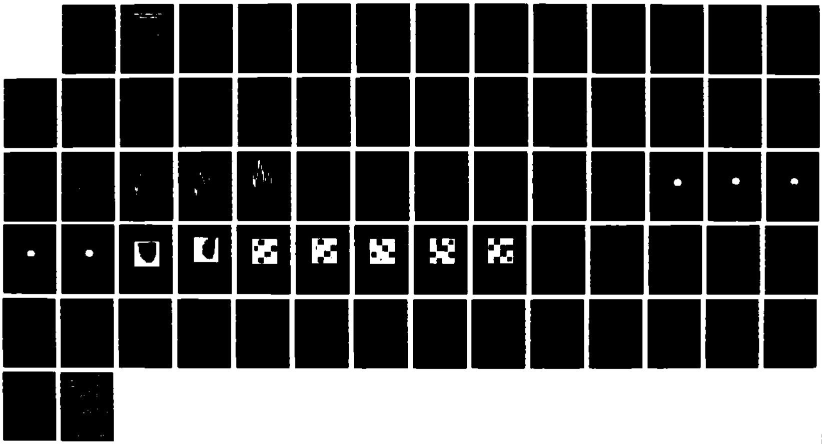
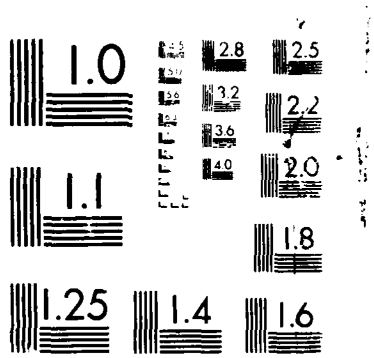
AD-A195 229 SEGMENTATION OF NOISY IMAGES USING NONSTATIONARY MARKOV 1/1
FIELDS(U) NAVAL POSTGRADUATE SCHOOL MONTEREY CA
K HACIPASAOGLU DEC 87

UNCLASSIFIED F/G 12/4 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963

DTIC FILE COPY (2)

NAVAL POSTGRADUATE SCHOOL
Monterey, California

AD-A195 229

DTIC ELECTE JUN 03 1988 S D H

THESIS

SEGMENTATION OF NOISY IMAGES USING NONSTATIONARY MARKOV FIELDS

by

Kani Hacipasaoglu

December 1987

Thesis Advisor Roberto Cristi

Approved for public release; distribution is unlimited.

88

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY 2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; Distribution is unlimited
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b. OFFICE SYMBOL (If applicable) 62	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING / SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)

SEGMENTATION OF NOISY IMAGES USING NONSTATIONARY MARKOV FIELDS

12. PERSONAL AUTHOR(S)
Hacipasaoglu, Kani

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Master's Thesis	FROM ______ TO ______	December 1987	

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Markov Random Field; Dynamic Programming; Kalman Filtering techniques.

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The purpose of this thesis is to develop an algorithm for segmenting images corrupted by a high level of noise with different characteristics. In particular the images considered are composed of several regions describing different objects and background. The algorithm described is based on a Markov Random Field (MRF) model of the image and uses Kalman Filtering ~~(KF)~~ techniques and Dynamic Programming (DP) in order to smooth within the regions. The theoretical background for one dimensional and two dimensional data which have different characteristics and simulation results are presented, with examples of synthetic data and underwater images.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION
22a. NAME OF RESPONSIBLE INDIVIDUAL Roberto Cristi	22b. TELEPHONE (Include Area Code) (408)646-2122 22c. OFFICE SYMBOL 62CX

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

U.S. Government Printing Office: 1986—606-243

Approved for public release; distribution is unlimited.

Segmentation of Noisy Images Using Nonstationary Markov Fields

by

Kani Hacipasaoglu
1st. Lieutenant, Turkish Army
B.S., Turkish Military Academy, 1980

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1987

Author: ______________________
Kani Hacipasaoglu

Approved by: ______________________
Roberto Cristi, Thesis Advisor

Ralph Hippenstiel
Ralph Hippenstiel, Second Reader

____
John P. Powers, Chairman,
Department of Electrical and Computer Engineering

____
Gordon E. Schacher,
Dean of Science and Engineering

ABSTRACT

The purpose of this thesis is to develop an algorithm for segmenting images corrupted by a high level of noise with different characteristics. In particular the images considered are composed of several regions describing different objects and background. The algorithm described is based on a Markov Random Field (MRF) model of the image and uses Kalman Filtering (KF) techniques and Dynamic Programming (DP) in order to smooth within the regions. The theoretical background for one dimensional and two dimensional data which have different characteristics and simulation results are presented, with examples of synthetic data and underwater images.

Accession For
NTIS GRA&I
By
Dist
A-1

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to express my deep thankfulness to my thesis advisor Assistant Professor R. Cristi for his numerous suggestions and help that have had a major influence on this thesis. I am indebted to the staff members of Computer Science Department for their help to run all my programs to simulate the algorithm of this study.

I am also grateful to all my teachers in the Naval Postgraduate School for the state-of-the art education. Finally, I wish to express my gratitude to my loving wife, Meral for the support, patience and understanding she provided throughout this thesis.

I. INTRODUCTION

The objective of segmentation is to divide a given image into meaningful regions or units. In many vision problems the first task is to distinguish objects to discriminate between various obstacles.

A general vision system for Artificial Intelligence applications can be decomposed in the following blocks:

1. *Data acquisition*: The picture of interest can be taken by suitable cameras for these purposes and can be digitized using proper software and hardware.
2. *Segmentation*: Removing noise and dividing the image into significant regions.
3. *Image understanding*: Deciding which objects are present and classifying them according to size, shape, etc.

The general approach at the basis of image segmentation is based on the identification of different properties characterizing the regions of interest. For example, several objects can be identified by their average intensity levels or by the textures on the surfaces. The task of an image segmentation algorithm is therefore to identify the regions from the vision signals which contain noise (due to the electronic equipment and other environmental factors such as turbid waters in underwater environments) or uninteresting details. For example, if it is desired to recognize the presence of a house in the picture, we need to segment the image by disregarding the doors, windows, cracks on the wall, etc.

Although all the methods in the literature for segmentation are well suited to most of the cases of interest, they have poor performances in the presence of a high level of noise. In the cases of noisy data statistical techniques are more suitable. In particular these are based on classical techniques of estimation, such as Maximum Likelihood (ML) or Maximum a Posteriori (MAP). Statistical estimation techniques rely on the use of distinct statistical models for the original image and for the disturbances.

In this thesis we present a segmentation algorithm for images affected by a high level of noise, based on statistical techniques. In particular we use Markov Random Fields (MRF) as a model of the original image, and we assume White Gaussian noise as a model for the disturbance. The reason for the choice of MRF is based on several considerations:

1. They extend one dimensional well-known models to the multidimensional case.

2. They relate global statistics to local dependencies of the data.
3. They can model compact, distinct regions by a suitable choice of parameters.

The smoothing is obtained by a combination of MRF (for the transition between regions), Kalman Filtering Techniques (to filter within the regions) and Dynamic Programming (to determine the best sequence of edges which maximizes the likelihood function).

Related to the works of segmentation, the algorithms of Derin, Elliott and Cristi [Refs. 1,2], Geman [Ref. 3], Besag [Ref. 4] have shown the effectiveness of these techniques segmenting images corrupted by a high level of noise. Parallel to these works, a combination of Autoregressive and MRF models has been presented by Therrien [Ref. 5] to segment textured images of terrain data.

In the next chapter, the properties of modeling the original scene by Markov Random Field and choice of the parameters in Markov Random Fields are presented. Estimation algorithms for one and two dimensional data are given in Chapter III. Finally, simulation results for different characteristics of data are the subjects of Chapter IV.

II. STATEMENT OF THE PROBLEM AND MODELING USING MRF

The problem to be addressed in this chapter is how to model the original scene and its estimation from the given (observed) data. As stated in the Introduction the model for the original scene is assumed to be a Markov Random Field (MRF) whose properties are discussed in the next section. As explained below, the estimation is based on a Bayesian approach and a Maximum a Posteriori (MAP) techniques.

A. PROBLEM STATEMENT

Let the original scene $X = \{X_{k,t}\}$ be a random field on a finite two dimensional lattice $L = \{(k,t) \in Z^2 : 1 \leq k \leq N_1, 1 \leq t \leq N_2\}$ where Z is the set of integers, assuming discrete values $F_1, F_2, F_3, \ldots$, which are constant in regions $R_1, R_2, R_3, \ldots$ with $R_i \subset L$ and $R_i \cap R_j = 0$ for $i \neq j$. For example this might be the case of several objects with different intensity levels.

Also let X be corrupted by an additive noise and modeled by a random field $W = \{W_{k,t}\}, (k,t) \in L$. Furthermore, W will be assumed to be a White Gaussian field with zero mean and known variance σ^2. W is identically independently distributed (i.i.d.) and independent of X.

Therefore, the observed image can be given by a random field $Y = \{Y_{k,t}\}$, as

$$Y_{k,t} = g_{k,t}(X_{k,t}, W_{k,t}) \tag{2.1}$$

where $(k,t) \in L$ and $g_{k,t}(\cdot,\cdot)$ is an arbitrary function. In our case we assume additive noise, and therefore the observed signal can be expressed as

$$Y_{k,t} = X_{k,t} + W_{k,t} \tag{2.2}$$

Now, the problem is to estimate X from the data Y. Since we assume that only noisy observations are available, the estimate can be obtained by maximization of the a posteriori distribution of the scene with respect to x by a Bayesian approach. Therefore

$$P_{X|Y}(\hat{x}|y) = \max_{\{x\}} P_{X|Y}(x|y) \tag{2.3}$$

where $\hat{x}$ is the estimate of original scene and $P_{X|Y}(x|y)$ = Probability $(X = x | Y = y)$. As it can be seen from Equation 2.3, the problem is to maximize the a posteriori probability of $\hat{x}$ given y. This form of Bayesian estimation is known as *Maximum a Posteriori* or MAP estimation.

Bayes factorization and the convenience of logarithmic operation yields

$$\ln P_{Y|X}(y|\hat{x}) + \ln P_X(\hat{x}) = \max_{\{x\}} \{\ln P_{Y|X}(y|x) + \ln P_X(x)\} \quad (2.4)$$

The two likelihood terms on the left and right hand sides of Equation 2.4 are determined by the model of X and of the disturbance. The model for the scene X is assumed to be a Markov Random Field (MRF) which is formally defined by Besag [Ref. 3: p. 724] and by Elliot, Derin, Cristi and Geman [Refs. 6,7].

B. MARKOV RANDOM FIELDS (MRF)

Definition: Let L be a finite lattice $L = \{(k,t) : 1 \le k \le N_1, 1 \le t \le N_2\}$ and $\eta_{k,t}$ defined as a subset of L so that $(k,t) \bar{\in} \eta_{k,t}$ where $(k,t) \in L$ and $(i,j) \in \eta_{k,t}$ if and only if $(k,t) \in \eta_{i,j}$. Then a random field $X = \{X_{k,t}\}$ with the property

$$\begin{aligned} &P\{X_{k,t} = x_{k,t} \mid X_{i,j} = x_{i,j}, (i,j) \in L\} = \\ &P\{X_{k,t} = x_{k,t} \mid X_{i,j} = x_{i,j}, (i,j) \in \eta_{k,t}\} \end{aligned} \quad (2.5)$$

is a Markov Random Field with neighborhood $\eta_{k,t}$.

A Markov Random Field can be illustrated as in Figure 2.1, where the statistics of the element (k,t) indicated by a "*" depends on its neighbors indicated by a "O" only. Two simple neighborhoods are shown in Figure 2.2. These are:

$$\eta_{i,j}^{1} = \{(1,n) : 0 < (i-1)^2 + (j-n)^2 \le 1\} \quad (2.6)$$

and

$$\eta_{i,j}^{2} = \{(1,n) : 0 < (i-1)^2 + (j-n)^2 \le 2\} \quad (2.7)$$

Relatively simple neighborhood systems as in Figure 2.2 are adequate in modeling most scenes of interest.

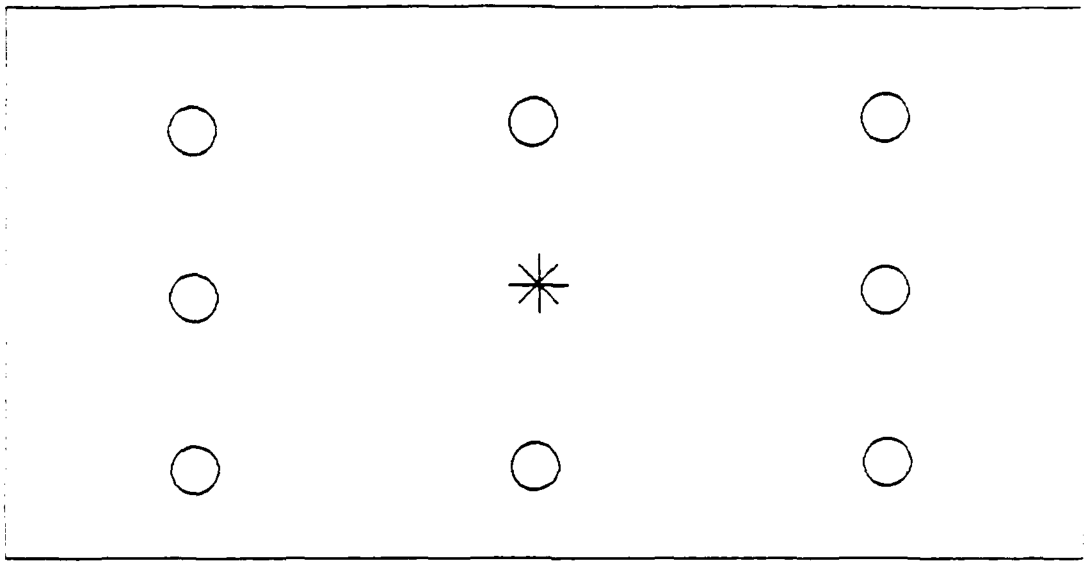


Figure 2.1 Dependence of Markov Random Field $P_X(*|\text{Everything else})$.

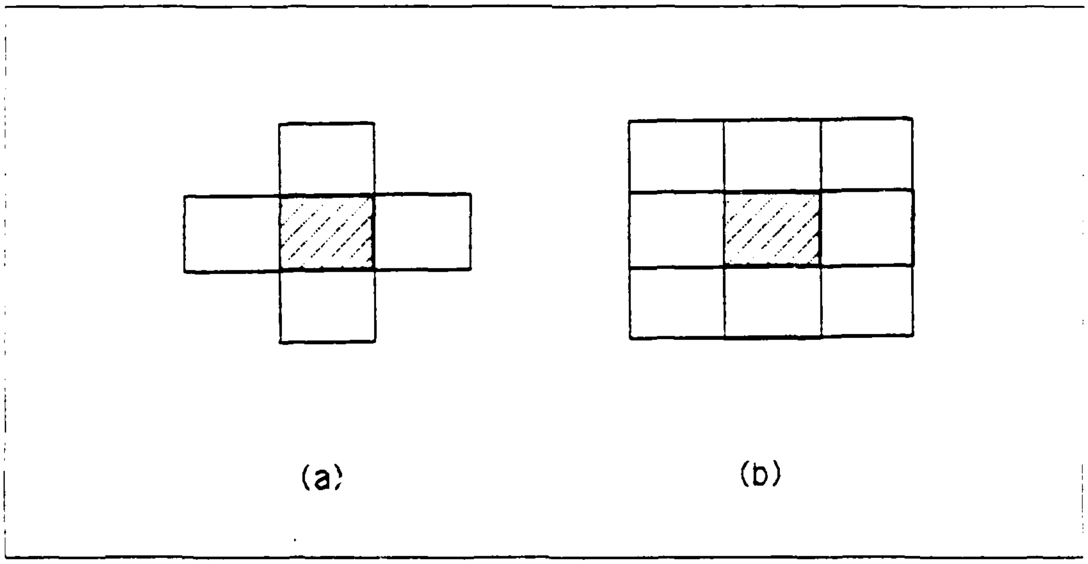


Figure 2.2 Neighborhoods $n_{i,j}^{1}$ (a) and $n_{i,j}^{2}$ (b).

Although a MRF can be considered as a multidimensional extension of a Markov Chain, a difficulty exists in defining MRF. In fact the main concept at the basis of Markov Chains is the concept of transition probabilities. These are subjects to

mild conditions and they rely on the fact that an ordering can be determined in one dimensional problems. This is not the case in multidimensional problems where an ordering of points in general does not exist, which presents a difficulty in the definition of transition probabilities. Because of this we have to approach MRFs in a different way from the Markov Chains. Therefore, to determine the structure for the joint probability of MRF models, we need some new definitions and theorems. In particular the joint probability is based on the concept of *clique* given below:

Definition: Given a Markov Random Field (MRF) with neighbors η a *clique* is a set of pixels which are neighbors of each other [Ref. 6: p.198]. The various types of cliques for $\eta_{i,j}^1$ and $\eta_{i,j}^2$ are shown in Figure 2.3. Based on the definition of cliques, the joint probability of a Markov Random Field with neighborhood η can be expressed by the following theorem:

Theorem (Hammersley and Clifford) [Ref. 6: pp. 192-199]. Suppose X is an MRF such that $P_X(x) > 0$ for all x, with neighborhood η. Then $P_X(x)$ can be defined as following:

$$P_X(x) = \frac{1}{Z} e^{-U(x)} \tag{2.8}$$

where

$$U(x) = \sum_{c \in \zeta} V_c(x) \tag{2.9}$$

ζ is the set of all cliques as shown in Figure 2.3 and usually $U(x)$ is defined as the energy function, $V_c(x)$ the potential associated with clique c, and the partition function Z is defined as

$$Z = \sum_{x} e^{-U(x)} \tag{2.10}$$

which is a normalizing constant that causes P_X to be a consistent probability measure. On the basis of the above theorem we can arbitrarily assign a joint probability of MRFs by the potential functions $V_c(x)$. The only constraint on $V_c(x)$ is that it has to

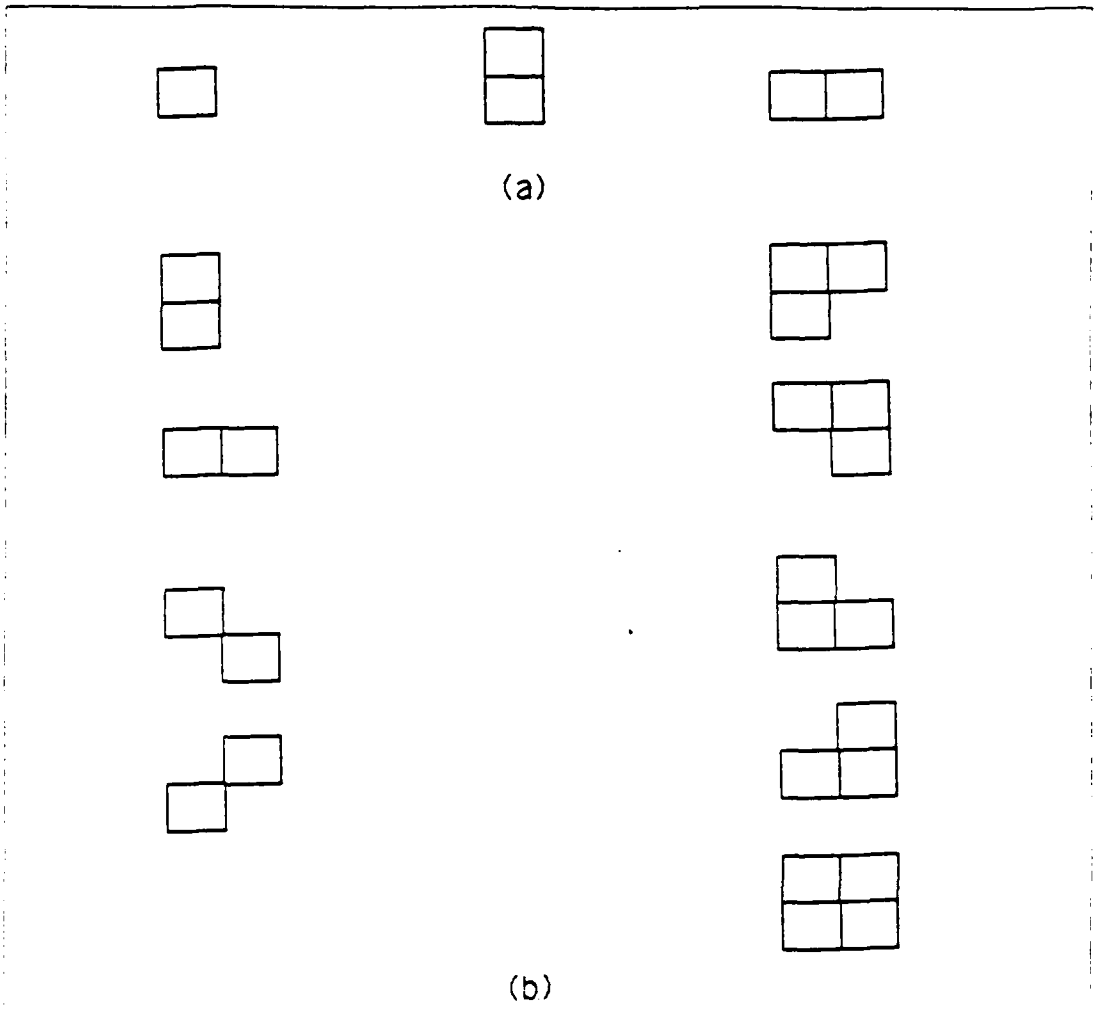


Figure 2.3 Cliques for Neighborhood Systems $\eta_{i,j}^{1}$ (a) and $\eta_{i,j}^{2}$ (b).

be finite for all x. It means that the form of $V_c(x)$ is fixed by the structure of the lattice given by Kinderman [Ref. 8].

In order to understand the above result and definitions, the joint probability of any Markov Random Field with neighborhood $\eta_{i,j}^{1}$ can be written as

$$\ln P_X(x) = \sum_{k,t} v_1(x_{k,t}) + \sum_{k,t} v_2(x_{k,t-1}, x_{k,t}) + \sum_{k,t} v_3(x_{k-1,t}, x_{k,t}) - \ln Z \tag{2.11}$$

for the case in Figure 2.3 (a).

Definition: A MRF which has all nonzero probabilities as in Equation 2.11, namely, $P_X(x) > 0$, is called a *Gibbs Field* (GF). The Gibbs model can be used to model the spatial interactions between regions, in which a pixel and its neighbors have similar statistical properties. In other words, it models spatial continuity which characterizes the images we want to segment. Note that the structure of $P_X(x)$ is relatively simple for $\eta_{i,j}^1$ as in Equation 2.11. Its complexity increases considerably for larger neighborhoods. For that reason, in general only the lower sets $\eta_{i,j}^1$ and $\eta_{i,j}^2$ are considered practically applicable, and even in $\eta_{i,j}^2$ cliques with only one or two elements are taken into account. In addition, the Gibbs distribution might also be used to model textures. In our problem we just want to model the fact that the most likely scenes have regions which are clusters of pixels, and the Markov Random Field model is adequate for this purpose, provided by properly assigning the potential functions. A particular model we are going to consider in this thesis is the MRF with neighbors η^1 as in Figure 2.3 (a) and with potential functions defined as:

$$v_1(x_{k,t}) = 0 \tag{2.12}$$

$$v_2(x_{k,t-1}, x_{k,t}) = v_3(x_{k-1,t}, x_{k,t}) = \begin{cases} \beta \text{ if } x_{k,t} = x_{k,t-1} \\ -\beta \text{ otherwise} \end{cases} \tag{2.13}$$

for any value of k and t and β a positive constant. The larger the parameter β, the more the joint distribution P_X is peaked around smooth realizations. By this definition we can write the joint probabilities as

$$\ln P_X(x) = \beta\{\sum_{k,t} g(x_{k,t}, x_{k,t-1}) + g(x_{k,t}, x_{k-1,t})\} - \ln Z \tag{2.14}$$

where

$$g(x_{k,t},\ x_{k-1,t}) = \begin{cases} 1 \text{ if } x_{k,t} = x_{k-1,t} \\ -1 \text{ otherwise} \end{cases} \tag{2.15}$$

Also, by the assumption of Gaussian noise with zero mean and standard deviation σ, Equation 2.14 with Equation 2.4 yields the likelihood as

$$\ln P_{Y|X}(y|x) + \ln P_X(x) = -\frac{1}{2\sigma^2}\sum_{k,t}(y_{k,t}-x_{k,t})^2 + \beta\{\sum_{k,t} g(x_{k,t},x_{k-1,t}) + g(x_{k,t},x_{k,t-1})\} - \ln Z \qquad (2.16)$$

Hence, given the data $y = \{y_{k,t}\}$ in noise, the noise model $W(\Phi, \sigma)$ and the Gibbs field model Equation 2.14 with parameter β, the estimate of the original scene is given by $\hat{x} = \{x_{k,t}\}$ such that

$$\frac{1}{2\sigma^2}\sum_{k,t}(y_{k,t}-x_{k,t})^2 - \beta\{\sum_{k,t} g(x_{k,t},x_{k,t-1}) + g(x_{k,t},x_{k-1,t})\} \qquad (2.17)$$

is minimal.

In particular there is no need to undertake the difficulty of calculating the partition function Z since it is a normalizing constant. So, it was omitted in Equation 2.17. Approaches to minimization of Equation 2.17 can be based on relaxation [Ref. 3: pp. 730-732] or deterministic [Ref. 4: pp. 266-267] or Dynamic Programming [Ref. 2: pp. 44-45 and Ref. 8: pp. 12-13]. In the next chapter a different approach to minimization of Equation 2.17 based on Kalman Filtering techniques will be presented.

C. CHOICE OF THE PARAMETER IN THE MRF

The difficulty in using the Gibbs Distribution as a region model is to estimate the parameters of the model from specific realizations. Some methods have been used previously to estimate the model parameters. For example, Besag [Ref. 6: pp. 211-212] suggested the *coding method* where the parameters are determined from subsets of data. This requires solution of a set of nonlinear equations. Another example of the parameter estimation method is given by Derin and Elliot [Ref. 2: pp. 43-44] which uses standard linear, least-squares estimation. This has been proved to be effective in modeling textures by GFs. A different approach was used by Geman [Ref. 3: pp. 727-729] which defined a *simulated annihiling*, where stochastic relaxation was combined with monotonic increasing parameters.

For this thesis, the parameter β of the model in Equation 2.17 was set by trial and error until a reasonable filtering was reached. The optimum value of the parameter β is smaller for high values of signal to noise ratio. So, it is necessary to modify this parameter as the signal to noise ratio changes.

III. SEGMENTATION AND SMOOTHING OF ONE DIMENSIONAL AND TWO DIMENSIONAL SIGNALS

The content of this chapter is the segmentation algorithms for one dimensional and two dimensional signals. Following a *Maximum a Posteriori* approach, estimate of the original data is obtained by minimization of the cost function given by Equation 2.17. To provide this, a smoothing technique is devised which is based on a combination of Kalman Filtering (KF) and Dynamic Programming (DP). Dynamic Programming is used to determine the best sequence of edges which maximizes the likelihood function and a detailed algorithm is presented in Appendix A. Kalman Filtering is used to filter within the regions. This filtering technique is preferred because the Kalman Filtering is the best linear optimal estimator. If the noise in the data is assumed to be Gaussian, the Kalman Filter gives the minimum variance estimate of the original scene. In particular it evaluates the conditional mean of the original data given the past observations (measurements). If the assumption of Gaussian noise is removed, the KF yields the best linear estimate.

In the next subsection, the segmentation algorithm for a binary sequence (i.e., one which assumes only two levels) is presented and the result will be extended to general multilevel cases. Related computer programs are included in Appendix B.

A. BINARY SEGMENTATION USING DYNAMIC PROGRAMMING ALGORITHM

Suppose that it is desired to segment a binary sequence having logic levels "1" and "0" which are related to two intensity levels (gray levels) corrupted by an additive noise. The noise is considered as a zero mean Gaussian noise with a known standard deviation σ. In the general image processing extension, we can consider a logic "0" as a background and a logic "1" as an object. An example of the original sequence and the noisy sequence is given in the next chapter.

Let x_i be the logic values of the original data of intensity levels, which assigns real numbers F(0), F(1) to the regions denoted by logical zeroes and ones respectively. Then the noisy data can be given as

$$Y_i = F(x_i) + W_i \tag{3.1}$$

where $W_i \sim N(0, \sigma)$. We can define the signal to noise ratio (SNR) as

$$SNR = \frac{F(1) - F(0)}{\sigma} \tag{3.2}$$

In this section we assume to know the levels F(0), F(1) and the noise variance σ^2. Here, the assumption of F and σ known is not restrictive, since various algorithms exist in the literature which enable one to estimate means and variance of mixtures of Gaussian populations. We have seen in Chapter 2 that a MRF can be arbitrarily assigned by the potential function V. In particular we can model spatial continuity by assigning high probabilities to smooth signals. This can be achieved by assigning the likelihood function in the following form:

$$\ell(x_0, x_1, \ldots, x_{N_1}) = \beta \sum_{i=0}^{N_1} g(x_{i+1}, x_i) - \frac{1}{2\sigma^2} \sum_{i=0}^{N_1} |y_i - F(x_i)|^2 \tag{3.3}$$

where

$$g(x_1, x_2) = \begin{cases} 1 \text{ if } x_1 = x_2 \\ -1 \text{ otherwise} \end{cases} \tag{3.4}$$

The parameter β models the smoothness of the original data. In recursive form the likelihood function can be written as:

$$\ell_{k+1}(x_0, x_1, \ldots, x_{k+1}) = \ell_k(x_0, x_1, \ldots, x_k) + \beta g(x_{k+1}, x_k) - \frac{1}{2\sigma^2} |y_{k+1} - F(x_{k+1})|^2 \tag{3.5}$$

with $\ell_{-1} = 0$. The estimate $\hat{x}$ is obtained by maximizing ℓ over all possible realizations x_i. Applying Dynamic Programming(DP) algorithm in Appendix A, the maximum of ℓ_k is obtained. Results of segmentation of noisy binary image is given in Chapter 4.

B. SMOOTHING OF PIECEWISE CONSTANT SIGNALS IN ONE DIMENSION USING KALMAN FILTERING

The algorithm presented in the previous section for binary data is suitable for a small number of levels. Although it can be generalized to any number of levels, its complexity increases considerably and becomes unfeasable in multilevel situations. So, for one and two dimensional observations which have more than two intensity levels, we need to search for a different approach.

As mentioned in the first section of this chapter, a new algorithm which uses the Kalman Filtering techniques and Dynamic Programming is developed. In applications filtering is concerned with the extraction of signals from noise. If the observation and original scene are modeled by linear stochastic models, a solution to the general filtering problem can be obtained by use of the Kalman Filter.

Due to the assumed piecewise characteristics of the data, the realizations of Y can be modeled by the state-space models

$$x_{t+1} = x_t + v_t \tag{3.6}$$

$$y_t = x_t + w_t \tag{3.7}$$

where w_t is Gaussian zero mean i.i.d. with standard deviation σ, and v_t *is nonzero at* the edges between regions only. By defining the original data X in one dimension as $X = \{X_t, t \in L\}$, $L = \{1,2,\ldots,N\}$, the one dimensional Markov model corresponding to MRF has joint density

$$\ln P_X(x) = \beta \sum_{t \in L} g(x_t, x_{t-1}) - \ln Z \tag{3.8}$$

and the likelihood function to be maximized like Equation 2.16 is

$$\ell(x) = -\frac{1}{2\sigma^2} \sum_{t \in L} (y_t - x_t)^2 + \beta \sum_{t \in L} g(x_t, x_{t-1}) \tag{3.9}$$

where the first term on the right hand side of this equation comes from the noise and the second term is the potential defined before.

The Kalman Filter for the state-space model in Equation 3.6 and 3.7 is defined by the recursion Equation 3.11 as

$$\hat{x}_{t-1} = \hat{x}_t + K_{t+1}(y_t - \hat{x}_t) \tag{3.10}$$

where K_t is the Kalman gain. When an edge is detected the gain needs to be reinitialized at that point. To do this, define a binary sequence

$$m_t = v_t \tag{3.11}$$

which has a logic "1" at the edges between the regions and a logic "0" in the regions. Just to give an example, let's say $T_1 = [t_1, t_2 - 1]$ and $T_2 = [t_2, t_3 - 1]$ where T_1, T_2 are adjacent regions and $t_1 < t_2 < t_3$. We can define $v_{t1} = v_{t2} = v_{t3} = 1$ as shown in Figure 3.1. In general

$$v_t = \begin{cases} 1 \text{ if } t-1 \in T_k \text{ and } t \notin T_k \text{ for some } k \\ 0 \quad \text{otherwise} \end{cases} \tag{3.12}$$

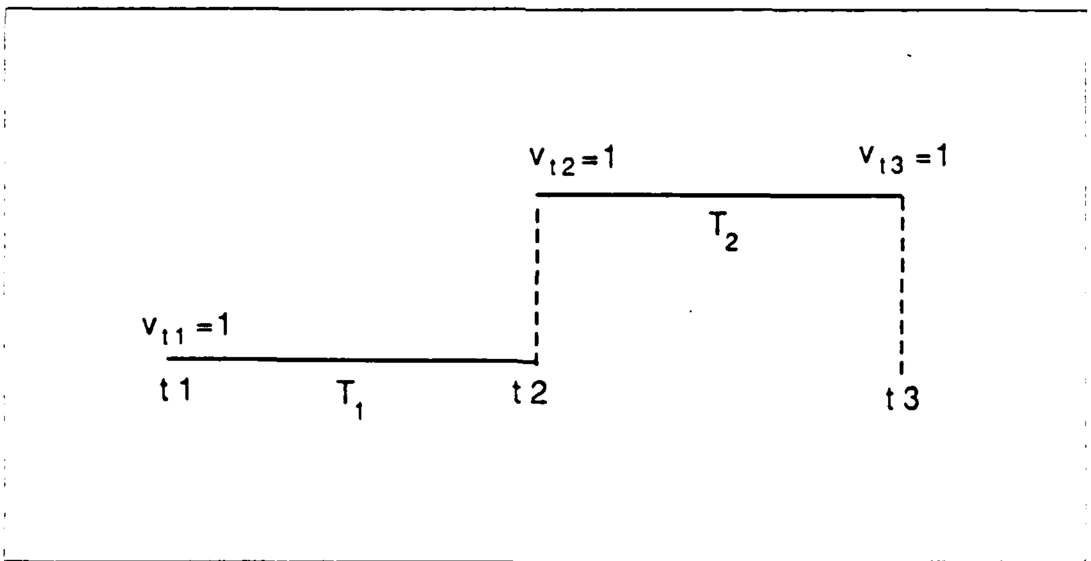


Figure 3.1 An Example for Adjacent Regions.

Then the estimated data can be determined from the noisy sequence y_t using Equation 3.10 and 3.11 as

$$\hat{x}_{t+1} = \hat{x}_t + K_{t+1}(y_t - \hat{x}_t) \tag{3.13}$$

where

$$K_t = \frac{1}{\tau_t} \tag{3.14}$$

and,

$$\tau_t = \begin{cases} 1 \text{ if } m_t = 1 \\ \tau_{t-1} + 1 \quad \text{otherwise} \end{cases} \tag{3.15}$$

Filtering Equation 3.13 yields the estimate of x as

$$x_t = \frac{1}{\tau_t} \sum_{j=1}^{\tau_t} y_{t-j} \tag{3.16}$$

where τ_t stands for the distance of t from the nearest left edge. Equation 3.16 expresses the Kalman Filter as an averaging of the measurements.

In the case that the edges are not known, observe the following:

1. Given the noisy data y for any binary sequence m, the estimate $\hat{x}$ is well defined by Equation 3.13 - 3.15, call this estimate $\hat{x}(m)$;
2. The g(. , .) term in the Markov model Equation 3.8 and the likelihood function Equation 3.9 can be interpreted as a penalty to the edges. From the dependence of x on the sequence m stated above it can be written.

$$g(\hat{x}_t, \hat{x}_{t-1}) = \begin{cases} 1 \text{ if } m_t = 0 \\ -1 \text{ if } m_t = 1 \end{cases} \tag{3.17}$$

Therefore we can define the function

$$\gamma(m_t) = \begin{cases} 1 \text{ if } m_t = 0 \\ -1 \text{ if } m_t = 1 \end{cases} \tag{3.18}$$

3. For any $j \in L$ let y^j, x^j, $\hat{x}^j$, m^j be the sequences and y, x, $\hat{x}$, m up to index j, for instance $y^j = \{y_0, y_1, \ldots, y_j\}$. With this definition the likelihood function Equation 3.9 associated with the estimate of x satisfies the recursion

$$\ell^{j+1}(m^{j+1}) = \ell^j(m^j) - \frac{1}{2\sigma^2}(y_{j+1} - \hat{x}_{j+1}(m^{j+1}))^2 + \beta\gamma(m_{j+1}) \tag{3.19}$$

In the case of the index set is $L = \{0,1,\ldots,N\}$ the likelihood function Equation 3.9 for the estimate $x(m)$ is

$$\ell(\hat{x}(m)) = \ell^N(\hat{x}(m)) \tag{3.20}$$

By these considerations the smoothing problem of the data is reduced to maximization of the likelihood function Equation 3.9 with respect to all binary sequences m. This can be done using the recursive formula Equation 3.19 and Dynamic Programming techniques given the details in Appendix A.

C. SMOOTHING OF PIECEWISE CONSTANT SIGNALS IN TWO DIMENSIONS USING KALMAN FILTERING

There is a problem in extending Kalman Filtering techniques from one dimension into two dimensions due to the lack of a causal state-space model for higher dimensions. Anyway, we can determine a two dimensional recursive formulation like Equation 3.13 - 3.15 by assuming the estimate $\hat{x}$ as the average value of the noisy data within the regions.

Proceeding just like the case in one dimension, a smoothing algorithm can be developed to maximize the likelihood function given below:

$$\ell(k,t) = -\frac{1}{2\sigma^2}\sum_{k,t}^{N}|y(k,t)-x(k,t)|^2 + \beta\sum_{k,t}^{N} g(x_{k,t}, x_{k,t-1}) + \beta\sum_{k,t}^{N} g(x_{k,t}, x_{k-1,t}) \tag{3.21}$$

To do this, first assume the edges to be known, and define $M = \{m_{k,t}\}$ where $m_{k,t} \in \{e_0, e_1, e_2, e_3\}$ for each $(k,t) \in L$, e_i being the four combinations of edges corresponding to the four possibilities in clique $\eta_{k,t}^{\ 1}$ as

$$g(x_{k,t}, x_{k,t-1}) = \pm 1 \tag{3.22}$$

and

$$g(x_{k,t}, x_{k-1,t}) = \pm 1 \tag{3.23}$$

These combinations of edges is shown in Figure 3.2.

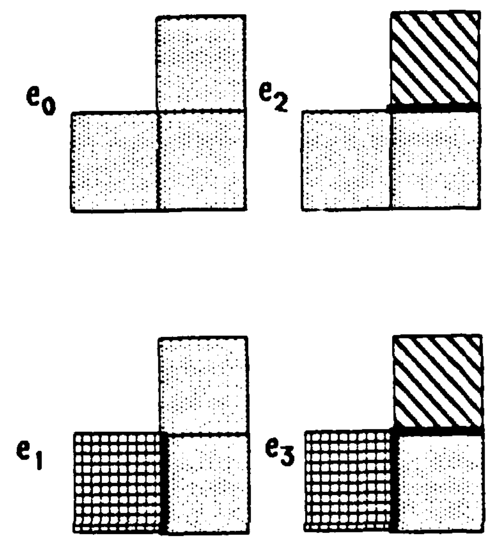


Figure 3.2 Illustration of Edges.

Some definitions are given below referring to Figure 3.3 for each point $(k,t) \in L$ to obtain the smoothing algorithm:

Definition 1: $\tau_{k,t}$ is the number of points between (k,t) and the closest edge above.

Definition 2: $\hat{z}_{k,t}$ is the average value of the noisy data y in the $(\tau_{k,t} \times 1)$ region.

Definition 3: $\lambda_{k,t}$ is the number of points in the homogeneous region surrounded by row k, column t and the line of edges.

Definition 4: $\hat{x}_{k,t}$ is the average of noisy data y in the region in which $\lambda_{k,t}$ is.

Using the definitions given above, the filtering algorithm can be developed as following

$$\hat{z}_{k,t} = \hat{z}_{k-1,t} + K_{k,t}(y_{k,t} - \hat{z}_{k-1,t}) \tag{3.24}$$

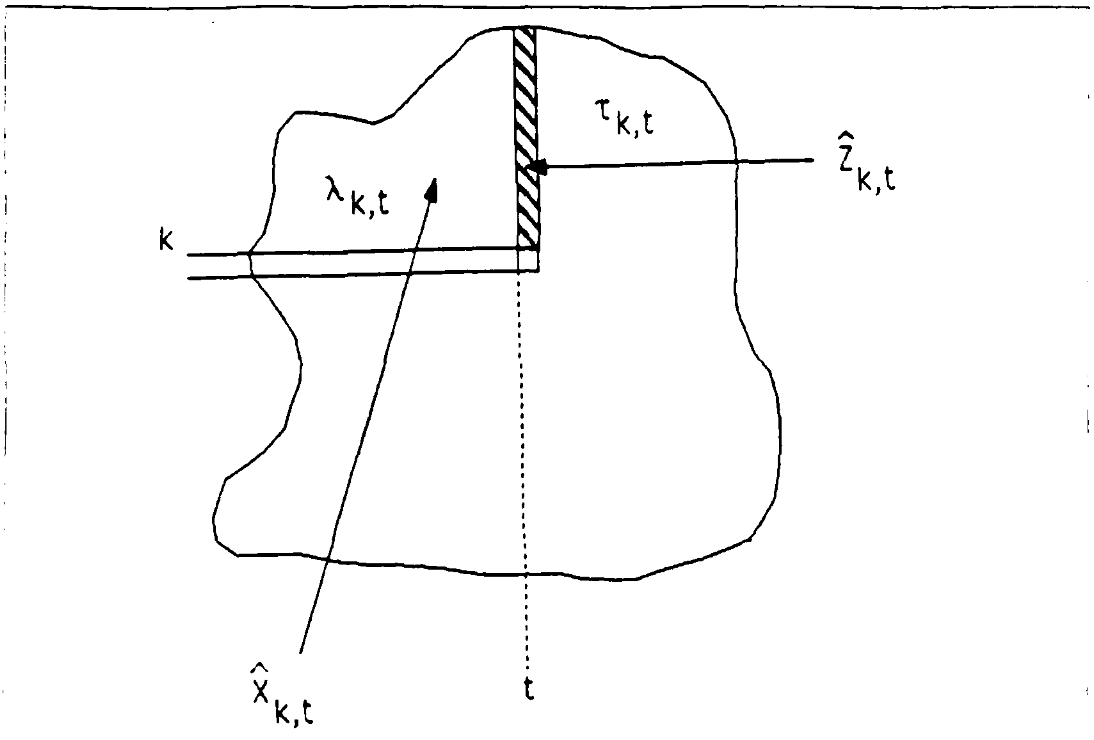


Figure 3.3 Illustration of Parameters of Two Dimensional Kalman Filter.

where

$$K_{k,t} = \frac{1}{\tau_{k,t}} \tag{3.25}$$

and

$$\hat{x}_{k,t} = \hat{x}_{k,t-1} + H_{k,t}(\hat{z}_{k,t} - \hat{x}_{k,t-1}) \tag{3.26}$$

where

$$H_{k,t} = \frac{\tau_{k,t}}{\lambda_{k,t}} \tag{3.27}$$

also

$$\tau_{k,t} = \begin{cases} \tau_{k-1,t} + 1 & \text{if } m_{k,t} \in \{e_0, e_1\} \\ 1 & \text{otherwise} \end{cases} \tag{3.28}$$

$$\lambda_{k,t} = \begin{cases} \lambda_{k,t-1} + \tau_{k,t} & \text{if } m_{k,t} \in \{e_0, e_2\} \\ \tau_{k,t} & \text{otherwise} \end{cases} \tag{3.29}$$

This algorithm is based on the fact that for any disjoint pair of regions $A, B \subset L$, the average of a noisy signal denoted by $\hat{x}$ on the regions A, B, $A \cup B$ are related as

$$\hat{x}(A \cup B) = a_1\hat{x}(A) + a_2\hat{x}(B) \tag{3.30}$$

with $a_1 = |A|/|A \cup B|$, $a_2 = |B|/|A \cup B|$ where $|\cdot|$ denotes the number of elements in the set. The recursion Equation 3.24 computes the average on A and Equation 3.26 the average on $A \cup B$. As in the one dimensional case Equations 3.21 - 3.24 represent a well-defined mapping which associates an estimate $\hat{x}(M)$ to any field of edges $M \in \{e_0,e_1,e_2,e_3\}^{N_1 \times N_2}$. Analogous considerations as for the one dimensional case lead to the likelihood function as

$$\mathcal{L}\{\hat{x}[m(v_{k,t})]\} = \mathcal{L}^{N_1 \times N_2}\{\hat{x}[m(v_{k,t})]\} \tag{3.31}$$

where

$$m(v_{k,t}) = \begin{cases} +1 & \text{if } v_{k,t} = e_0 \\ 0 & \text{if } v_{k,t} = e_1, e_2 \\ -1 & \text{if } v_{k,t} = e_3 \end{cases} \tag{3.32}$$

So, the estimate $\hat{x}$ can be determined by searching over all possible sets of edges M in order to find $\hat{x}$ which maximizes the likelihood function Equation 3.21. However the Dynamic Programming algorithm is used to maximize the likelihood function in two dimensions.

IV. RESULTS OF SIMULATIONS

A. ONE DIMENSIONAL SIGNALS

We started the simulations by creating a one dimensional two level binary sequence which has 128 points as shown in Figure 4.1. Gaussian zero mean noise with different signal to noise ratios (SNRs) calculated with the formula given by Equation 3.2 was generated and added to the original signal by using IMSL library subroutines. The noisy sequences are given in Figure 4.2, 4.3, 4.4 and 4.5. Here, the standard deviations for the noise are 25, 50, 75 and 100 respectively.

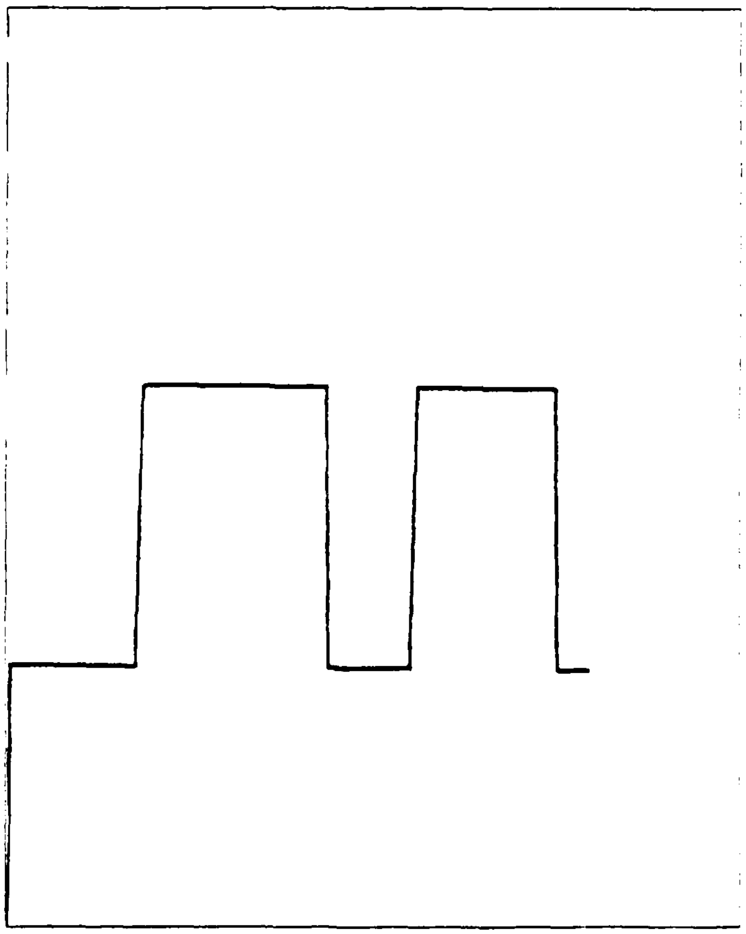


Figure 4.1 Original Binary Sequence.

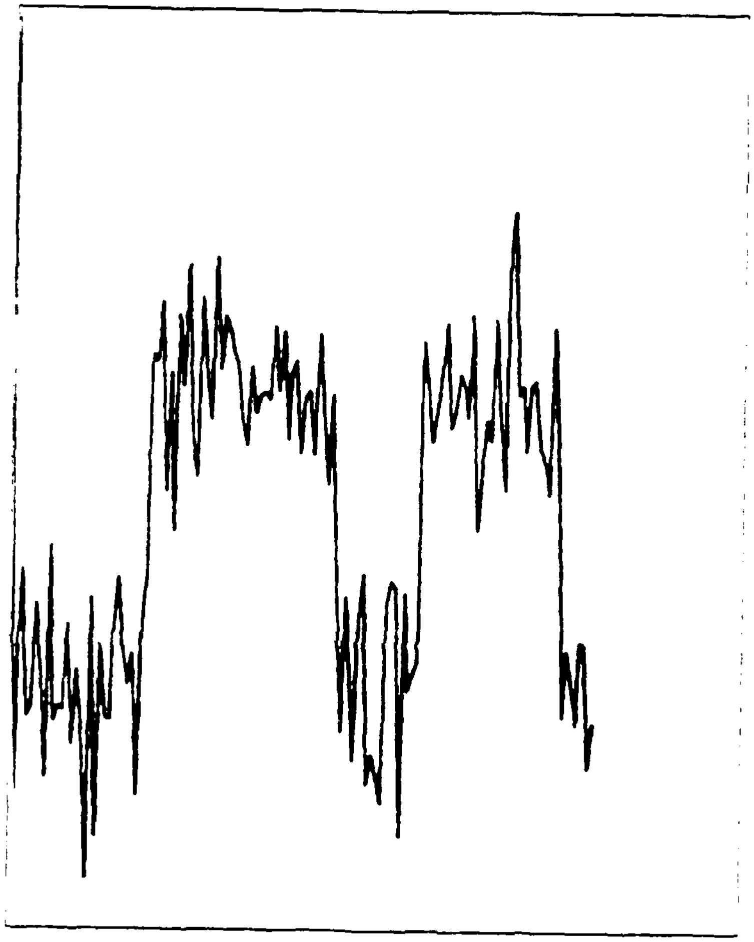


Figure 4.2 Noisy Sequence with $\sigma = 25$
SNR = 4.

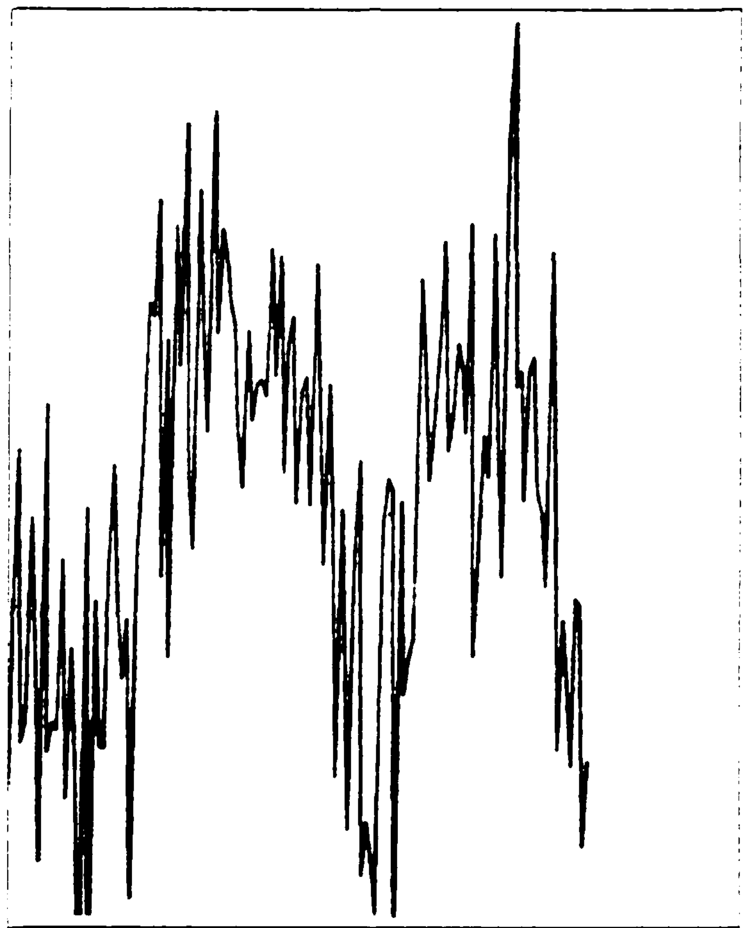


Figure 4.3 Noisy Sequence with $\sigma = 50$
SNR = 2.

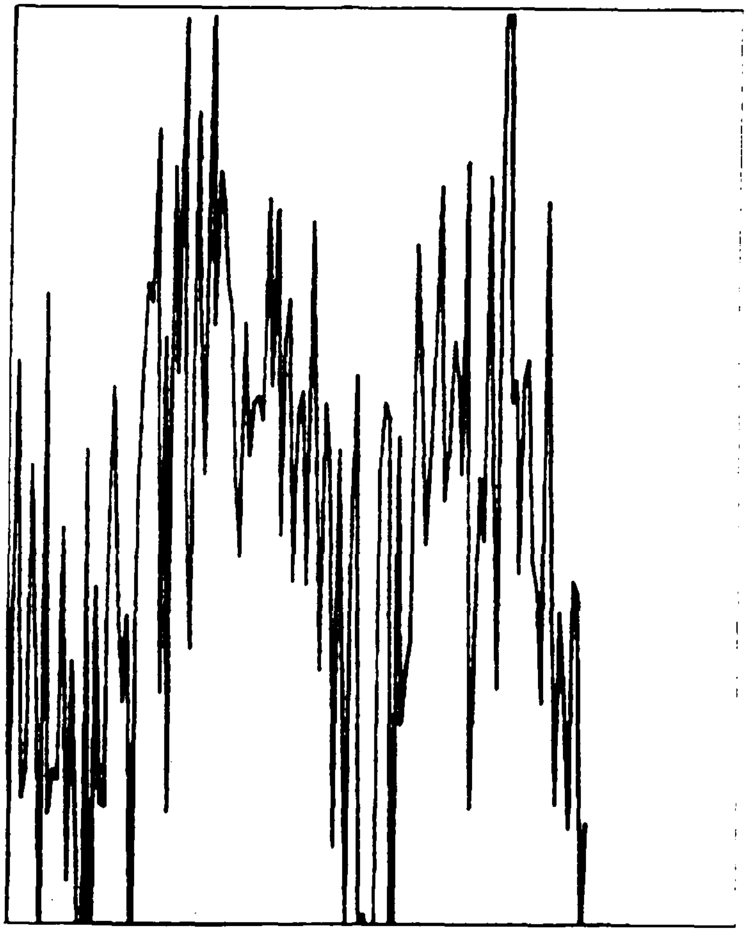


Figure 4.4 Noisy Sequence with $\sigma = 75$
SNR = 1.25.

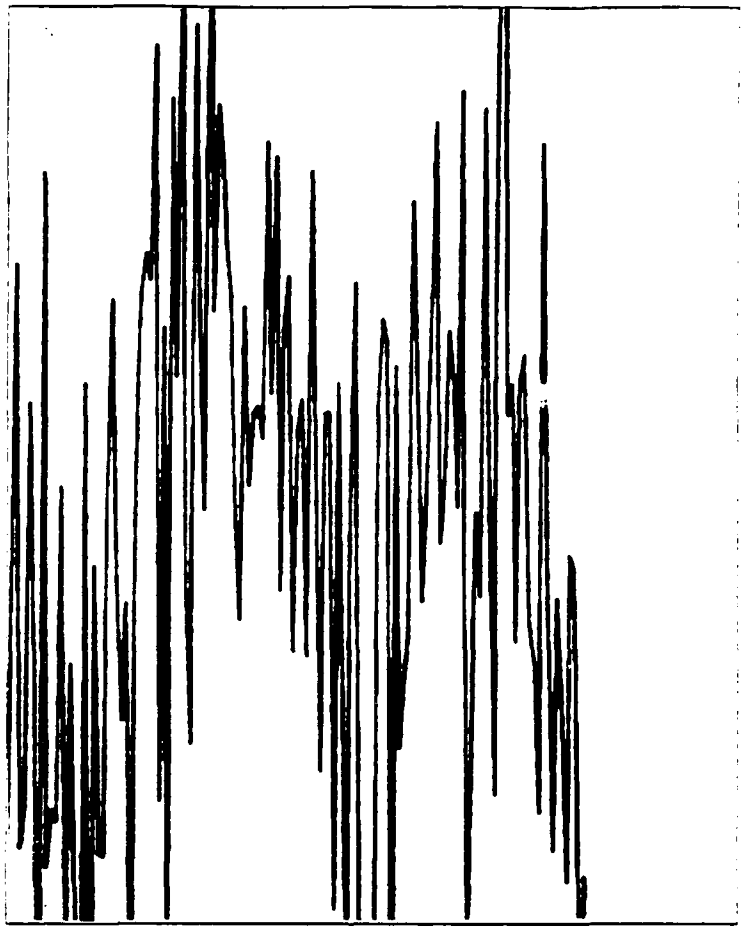


Figure 4.5 Noisy Sequence with $\sigma = 100$
SNR = 1.

The segmentation algorithm explained in the first section of Chapter 3 which uses Dynamic Programming was applied to filter out the noisy signal with the signal to noise ratios given above. To improve the result, for each simulation the algorithm was run twice, from left to right and from right to left in order to provide smooth results.

The outcome of simulations depend on the model parameter β that models the smoothness of the original data and this parameter was switched several times until we had the best segmentation for each case. The values of β were set by trial and error for this study. The best values for β for the given SNRs were obtained as 0.84, 0.80, 0.74 and 0.72 respectively. Filtered signals are presented in Figures 4.6, 4.7, 4.8 and 4.9.

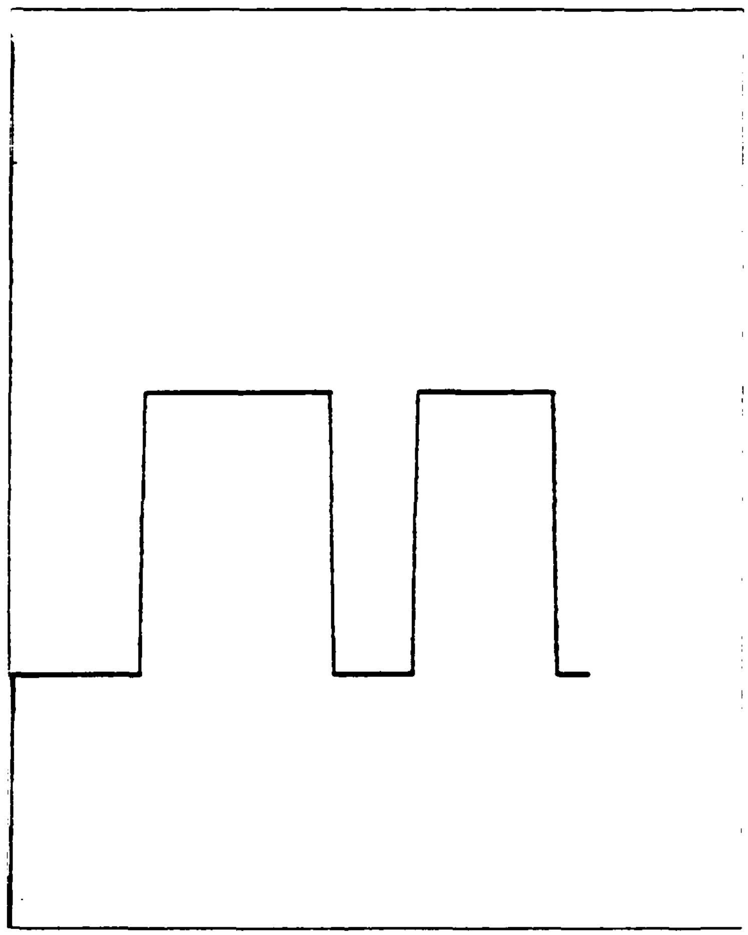


Figure 4.6 Segmented Sequence for $\sigma = 25$ and $\beta = 0.84$.

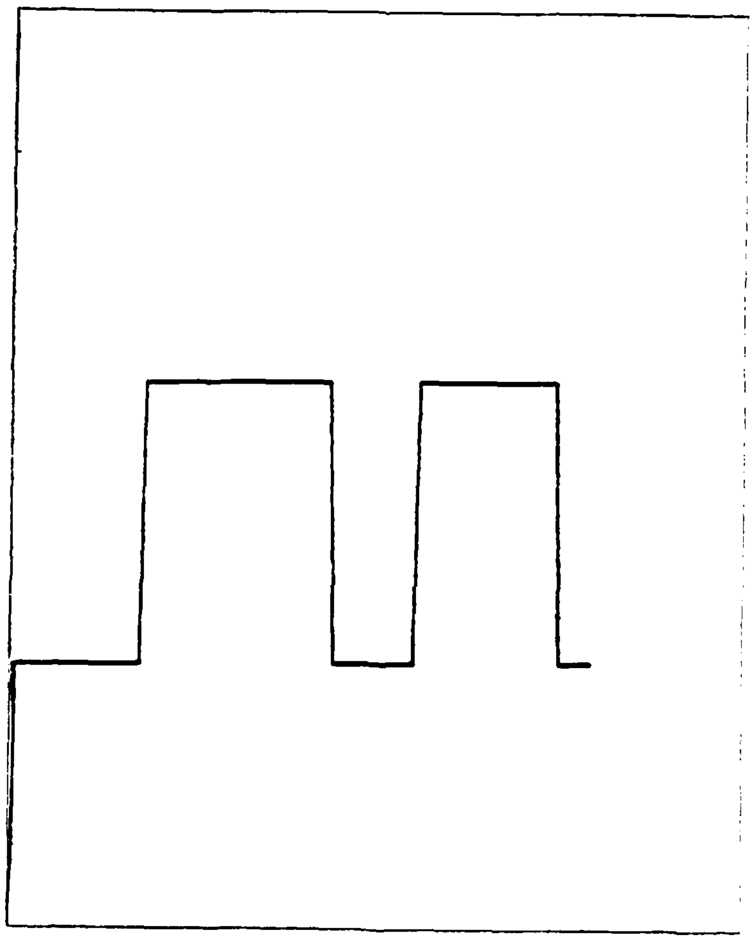


Figure 4.7 Segmented Sequence for $\sigma = 50$ and $\beta = 0.50$.

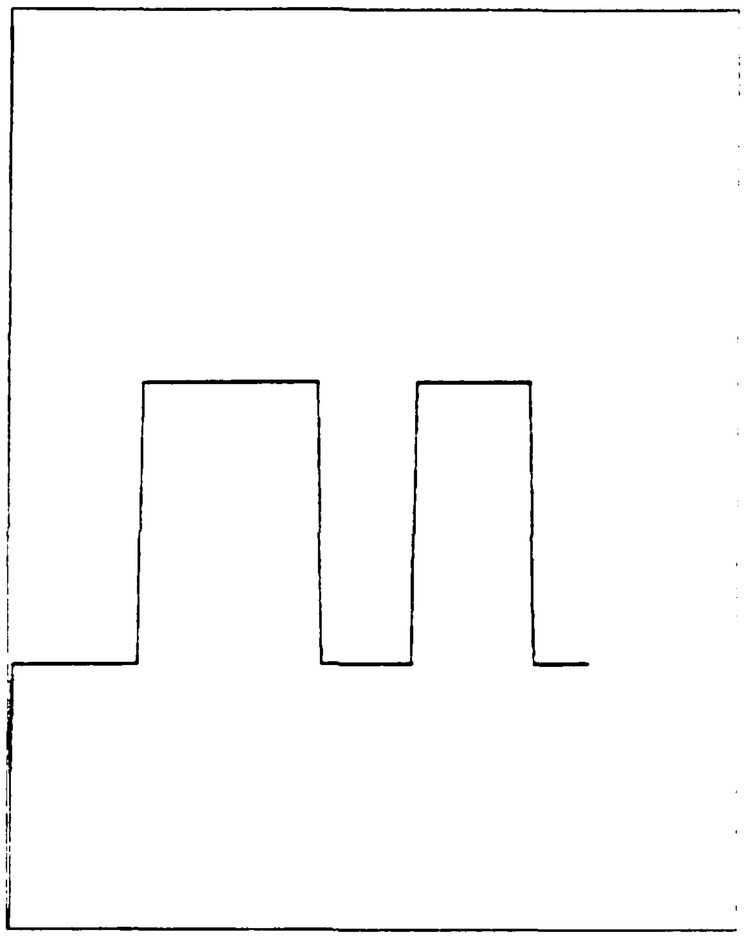


Figure 4.8 Segmented Sequence for $\sigma = 75$ and $\beta = 0.74$.

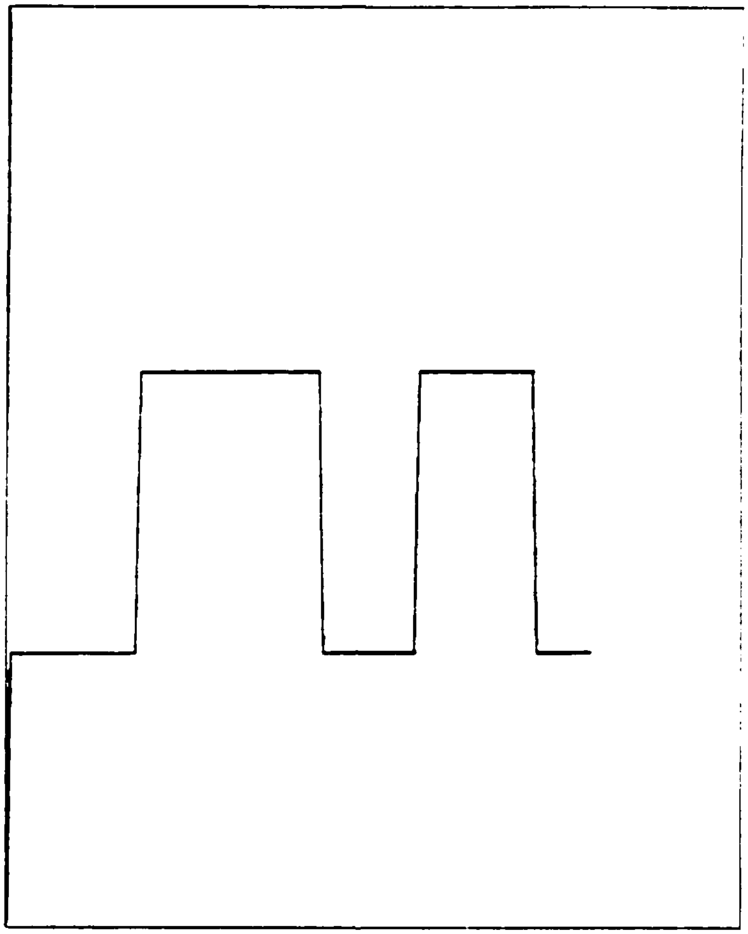


Figure 4.9 Segmented Sequence for $\sigma = 100$ and $\beta = 0.72$.

For the case of data with unknown levels we developed a different segmentation algorithm. The difficulty is to estimate the data levels, as well as to determine the segmentation. This segmentation algorithm is based on Kalman Filtering and Dynamic Programming. First, this method has been applied to the one dimensional noisy signals shown in Figures 4.2, 4.3, 4.4 and 4.5. The noisy data was filtered twice (two passes). The two passes results are shown in Figures 4.10, 4.11 for $\sigma = 25$ and 50.

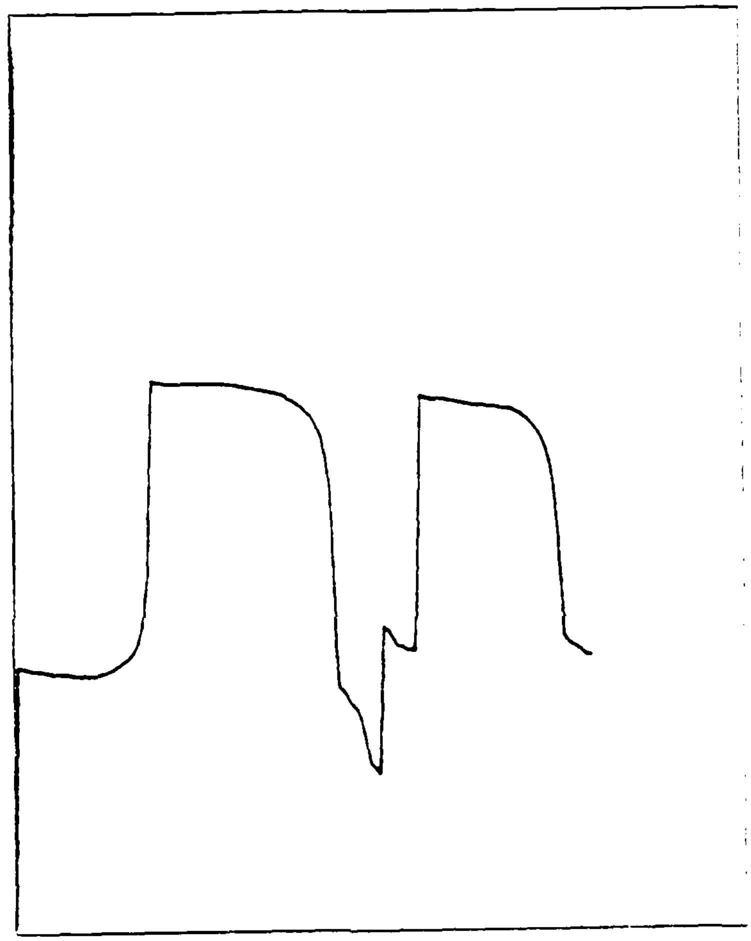


Figure 4.10 Segmented Sequence by KF with $\sigma = 25$.

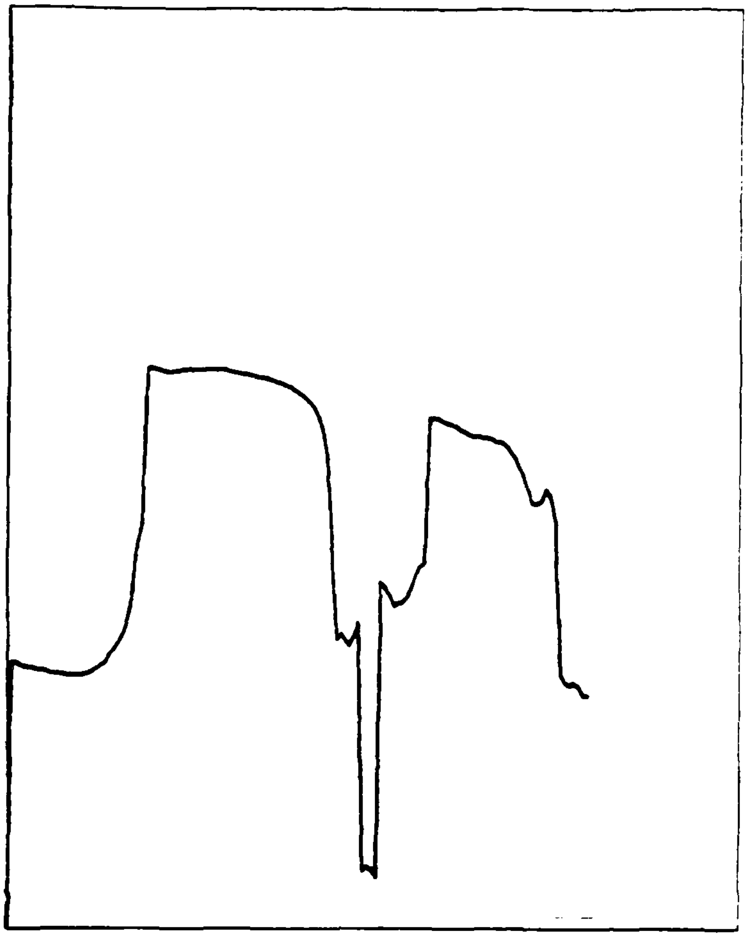


Figure 4.11 Segmented Sequence by KF with $\sigma = 50$.

B. TWO DIMENSIONAL IMAGES

We applied the two dimensional filtering algorithm to the test images which contain three different regions given in Figure 4.12. In this test image there are four different intensity levels (one for the background and three for the objects). Random Gaussian noise has been added with standard deviation $\sigma = 10$ and $\sigma = 20$ as shown in Figures 4.13, 4.14.

The result of filtering is shown in Figures 4.15 and 4.16. Notice that in these cases the noise has been removed while preserving the edges between the regions. The values of the parameter β giving the best results are $\beta = 2.0$ for $\sigma = 10$ and $\beta = 1.8$ for $\sigma = 20$ showing a trend of this algorithm. The value of β depends on the SNR of the noisy picture. In particular β should be decreased for higher levels of noise.

As an application to a test image in underwater environment, we applied the algorithm to a 512×512 picture of a fish corrupted by additive noise. The noisy image is shown in Figure 4.17 and the filtered one is shown in Figure 4.18. The significance of it is the fact that after filtering the fish and the background present well distinct intensity levels. Although applications to this class of problems are still under investigation, the fact that the object (the fish in this case) presents characteristics well distinct from the background can be used for automatic detection or recognition in an artificial intelligence framework.

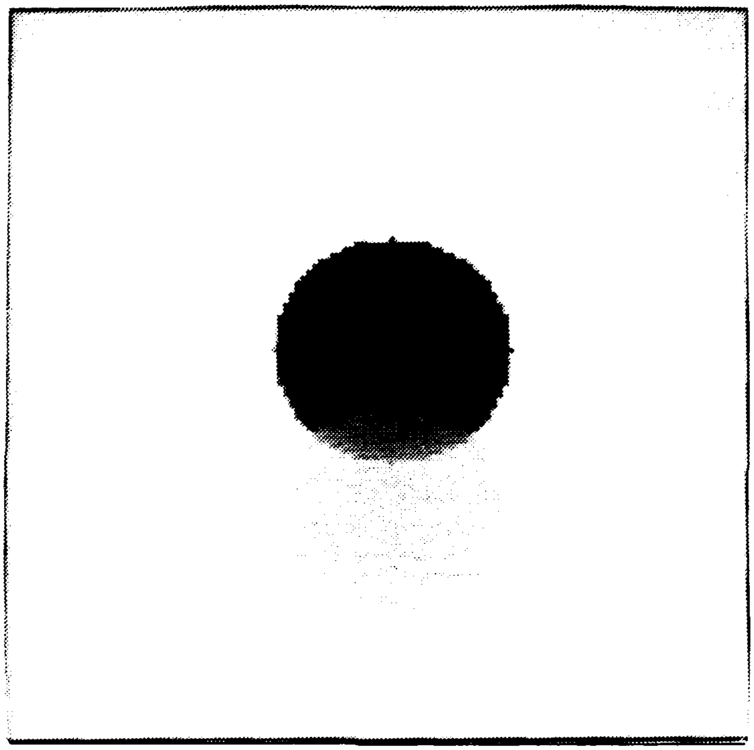


Figure 4.12   Original Test Image.

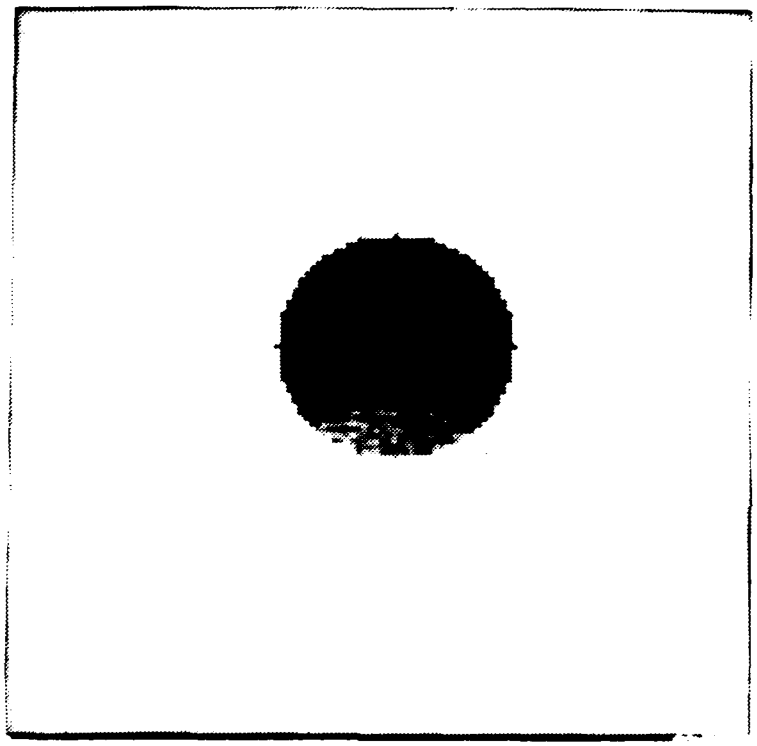


Figure 4.13 Test Image with $\sigma = 10$.

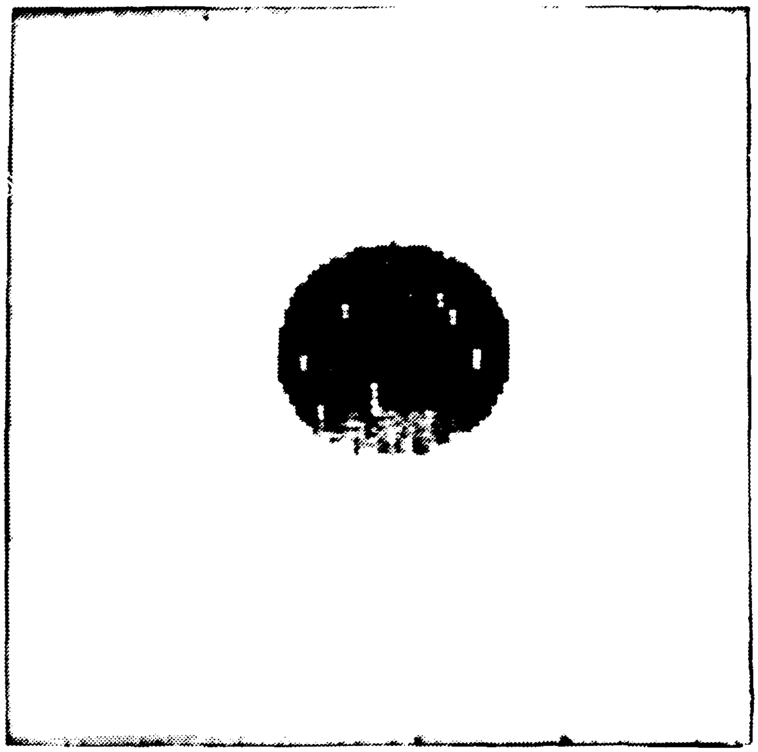


Figure 4.14 Test Image with $\sigma = 20$.

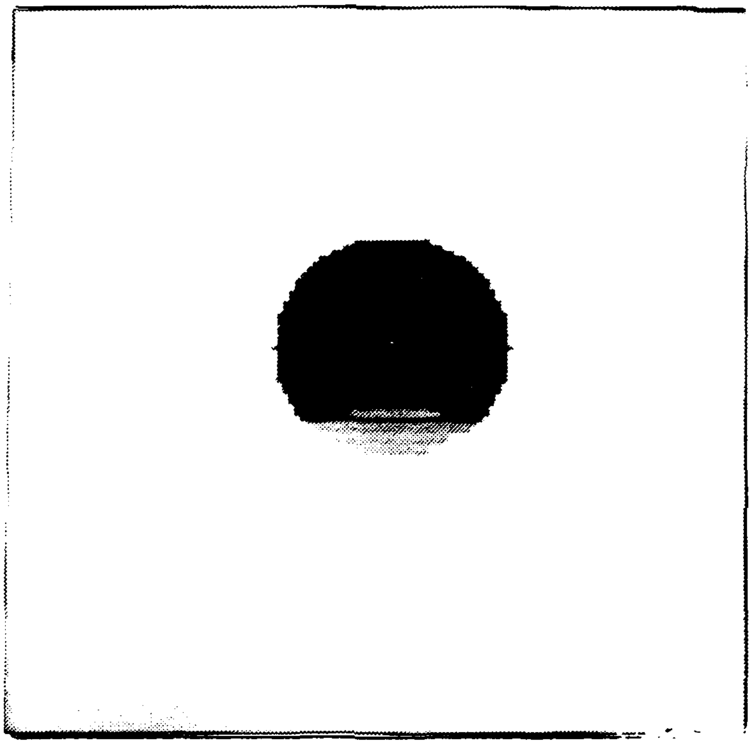


Figure 4.15 Segmented Image with $\sigma = 10$ and $\beta = 2.0$.

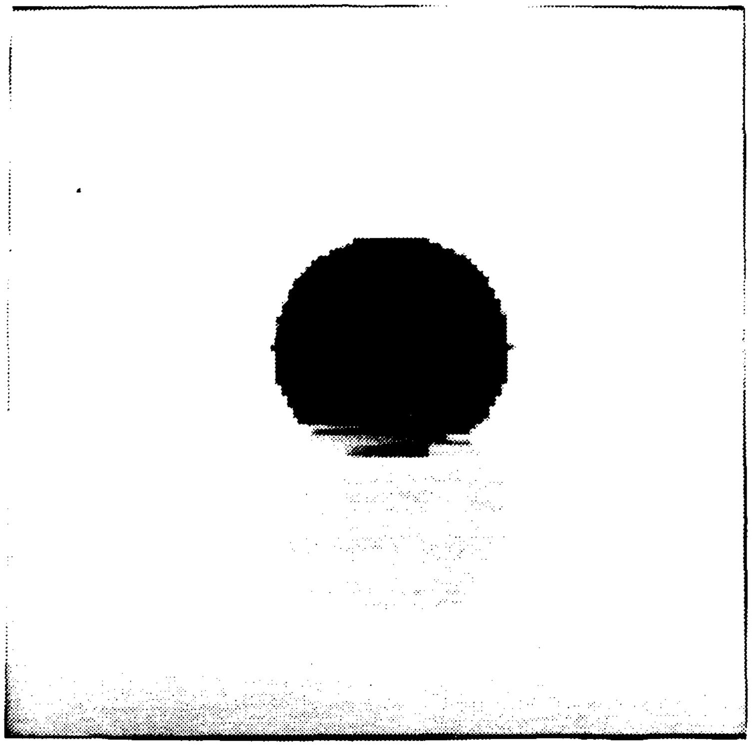


Figure 4.16 Segmented Test Image with $\sigma = 20$ and $\beta = 1.8$.

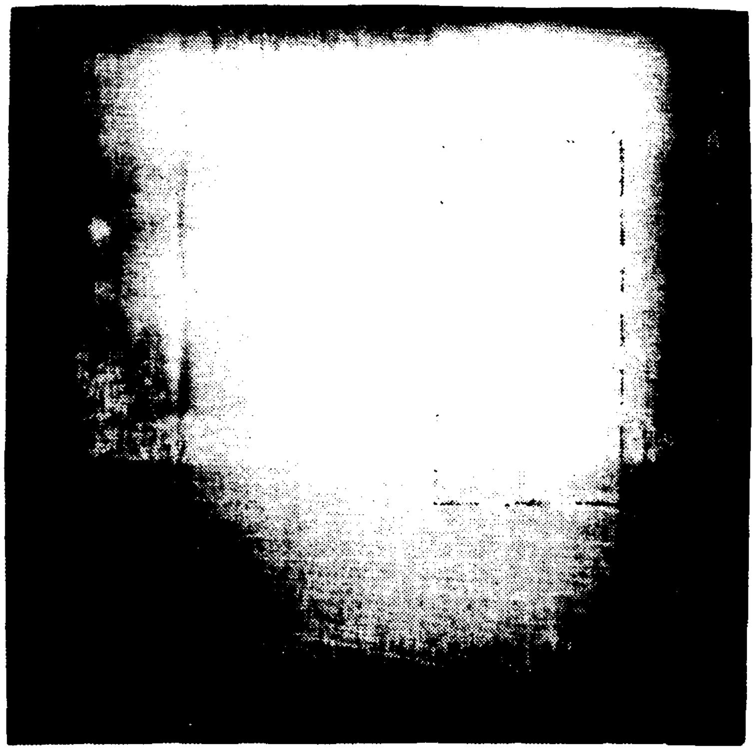


Figure 4.17 Image of Fish with $\sigma = 25$.

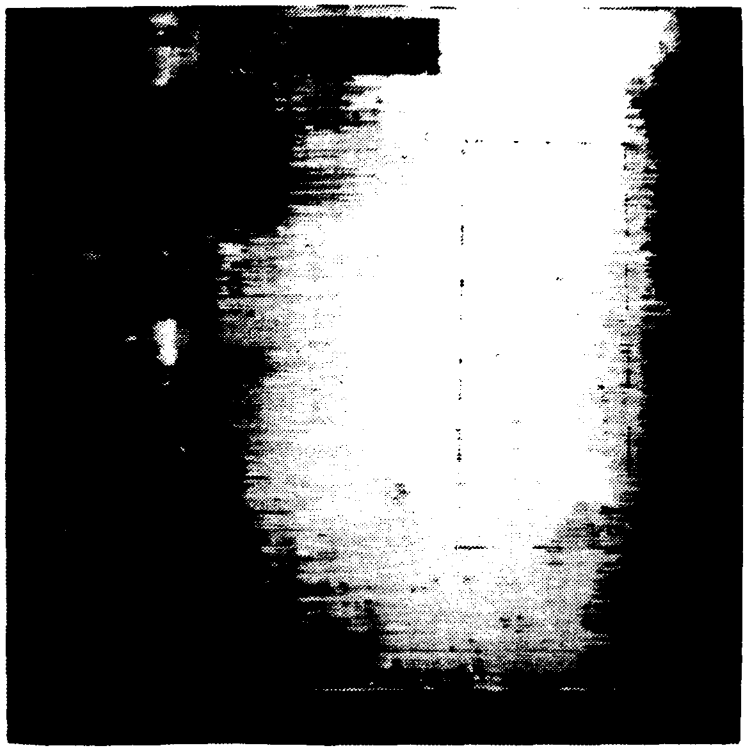


Figure 4.18 Segmented Fish with $\sigma = 25$ and $\beta = 0.25$.

Analogously we tested the algorithm on a 16 level checkboard shown in Figure 4.19 for different levels of noise. In particular the values of the 16 levels are given by 100, 50, 180, 70, 200, 120, 60, 140, 75, 175, 90, 65, 130, 55, 190, 110 from left to right for each line. The effects of additive noise are given in Figures 4.20 and 4.21 for $\sigma = 10$ and $\sigma = 20$ respectively. The noise is always assumed to be Gaussian, zero mean and white. The application of the filtering algorithm is shown in Figures 4.22 and 4.23 using $\beta = 1.2$ and $\beta = 0.7$ respectively. As expected the algorithm filters within the regions while preserving the sharp separation between adjacent regions. Also shown in Figure 4.22 are the *edges* between regions, as detected by the algorithm which shows the reinitializations of the Kalman Filter gains.

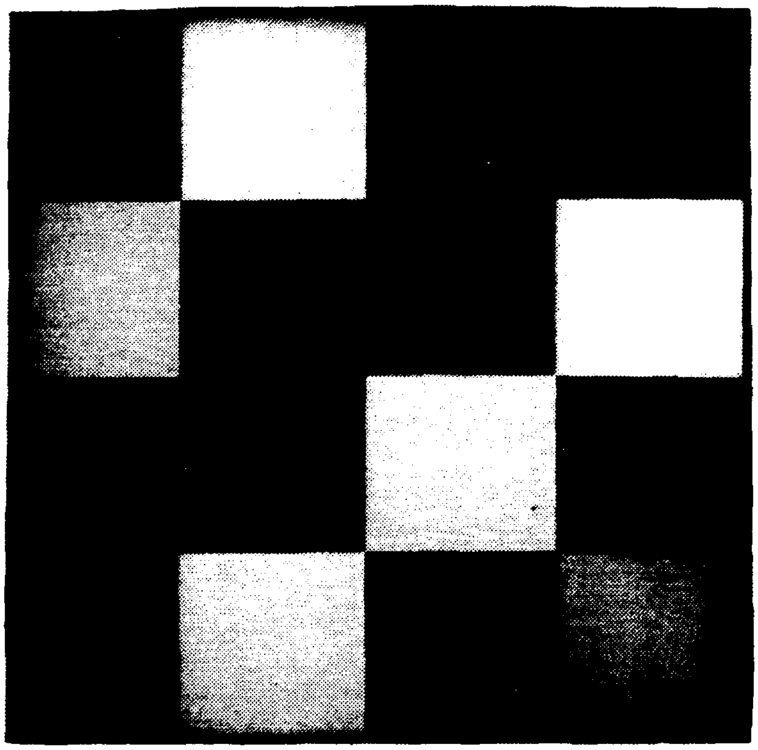


Figure 4.19 Test Image with 16 Different Regions.

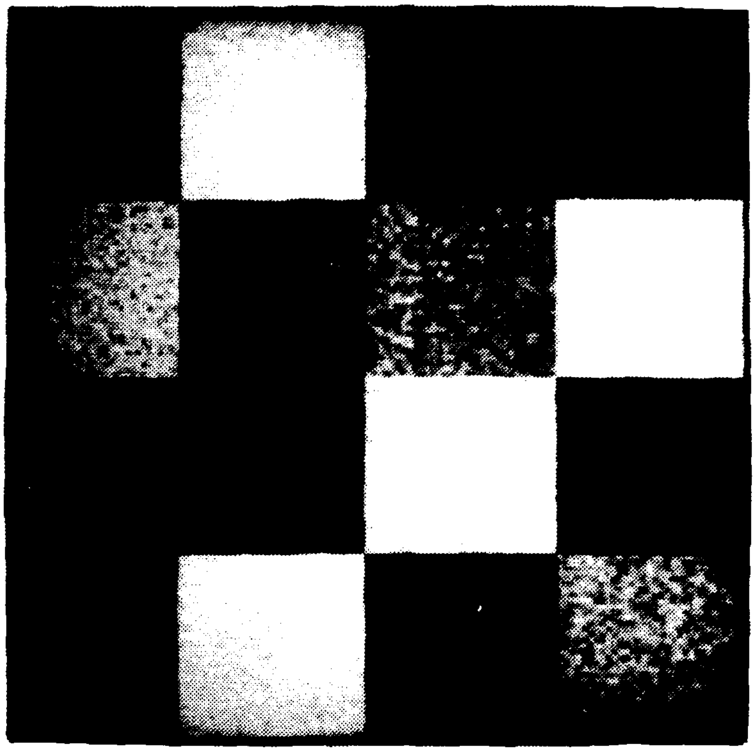


Figure 4.20 Test Image with $\sigma = 10$.

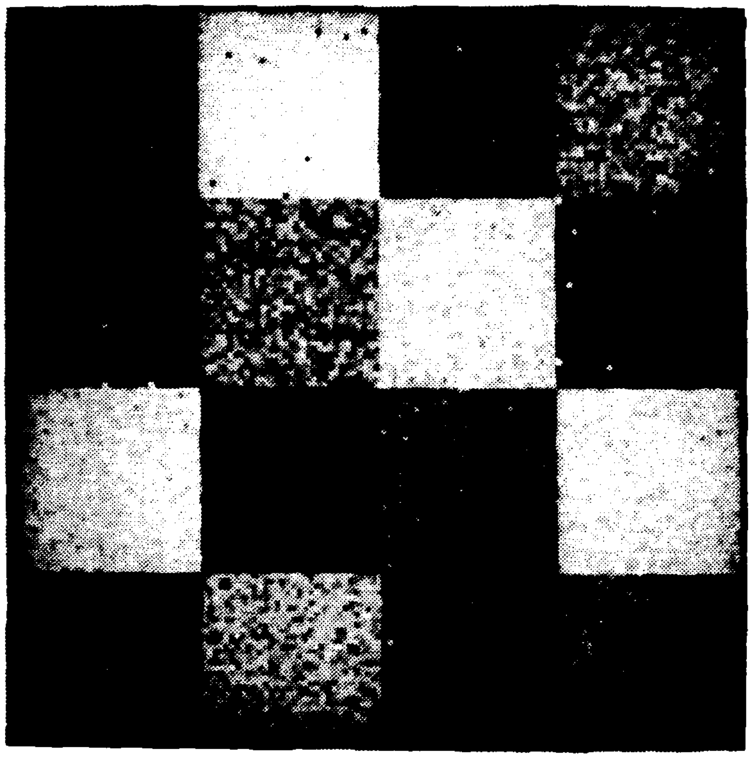


Figure 4.21 Test Image with $\sigma = 20$.

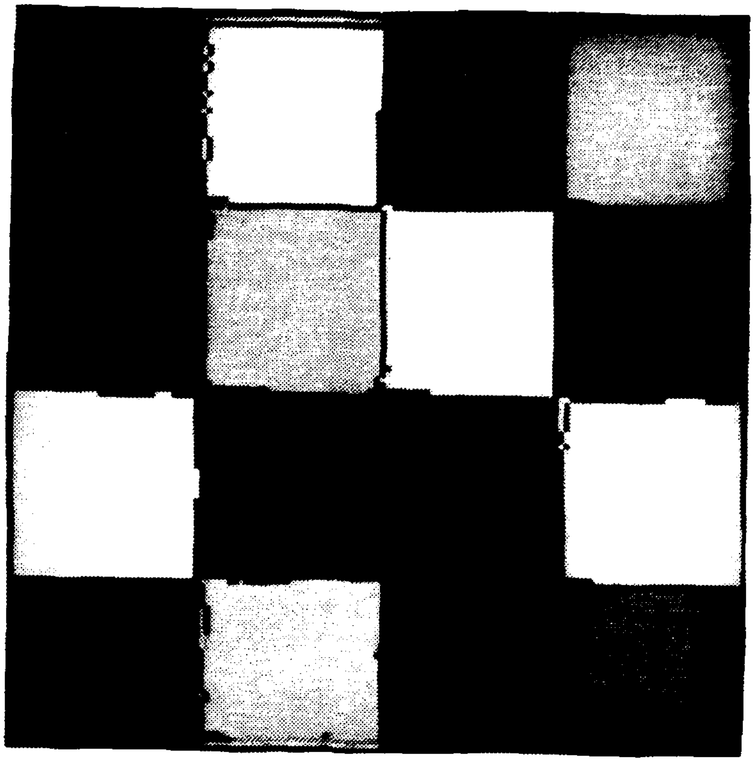


Figure 4.22 Segmented Image with $\sigma = 10$ and $\beta = 1.2$.

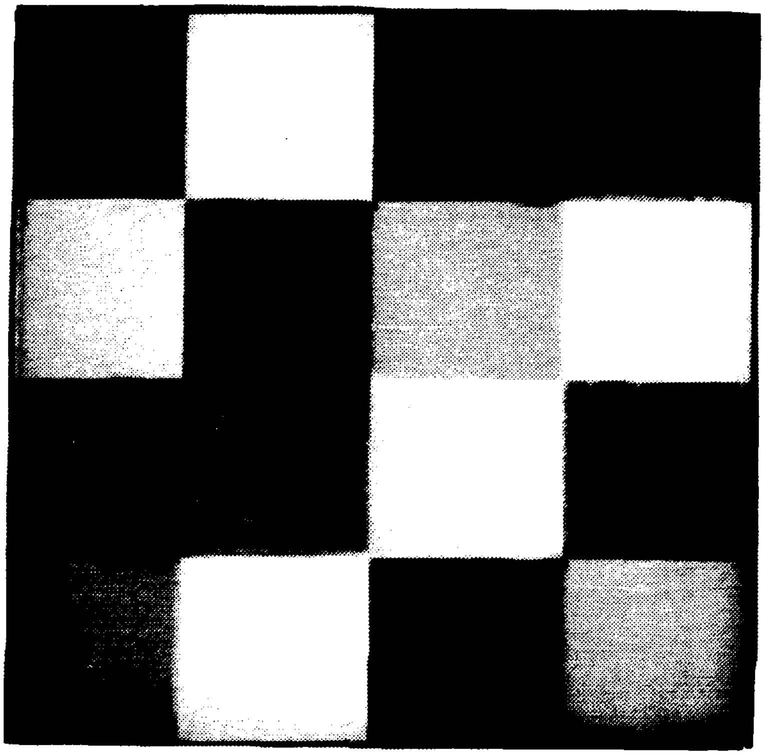


Figure 4.23 Segmented Image with $\sigma = 20$ and $\beta = 0.7$.

V. CONCLUSIONS

An algorithm to segment noisy data in one dimension and two dimensions has been presented, where the segments are piecewise constant. The data are described by state-space models. The particular feature of the algorithm is to search for the best sequence of edges in order to maximize the likelihood function. The algorithm has emphasized piecewise constant data, but it can be extended for the data with regions characterized by textures to which we associate different Autoregressive (AR) models. Typical applications are not only image segmentation but also segmentation in speech analysis, such as phenomenon separation.

APPENDIX A
DYNAMIC PROGRAMMING ALGORITHM

The Dynamic Programming is an algorithm used when the solution to a problem may be considered as the result of a sequence of decisions. An optimal sequence of decisions will maximize the given function, in the case that is used in this study, the likelihood function ℓ_k.

In Dynamic Programming an optimal sequence of decisions is obtained by making explicit appeal to the *Principle of Optimality*. In the cases when this principle can be applied, an optimal sequence of decisions has the property that whatever the initial state and decision are, the rest of the decisions must make up an optimal decision sequence according to the state resulting from the first decision. Standard Dynamic Programming techniques are introduced by Bellman. [Ref. 9]

The mathematical equations and the algorithm is given below for the case that was mentioned in the second section of Chapter 3. By defining the likelihood function as

$$\ell_k(x_0,x_1,\ldots,x_k) = \beta\sum_{j=0}^{k} g(x_j,x_{j-1}) - \frac{1}{2\sigma^2}\sum_{j=0}^{k}|y_j - F(x_j)|^2 \tag{A.1}$$

or in recursive form

$$\ell_{k+1}(x_0,x_1,\ldots,x_{k+1}) = \ell_k(x_0,x_1,\ldots,x_k) + \Delta\ell_{k+1}(x_{k+1},x_k) \tag{A.2}$$

where

$$\Delta\ell_{k+1}(x_{k+1},x_k) = \beta g(x_{k+1},x_k) - \frac{1}{2\sigma^2}|y_{k+1} - F(x_{k+1})|^2 \tag{A.3}$$

and setting the initial condition $x_{-1} = 0$ and also assuming that x_k can be only logic "0" or "1", we can determine the best sequence $\{\hat{x}_k\ ,\ k=0,\ldots,N\}$ which maximizes $\ell_N(x_0,\ldots,x_N)$. For this purpose define

$$J_k(0) = \max_{(x_0,\dots,x_{k-1})} \ell_k(x_0,\dots,x_{k-1},0) \tag{A.4}$$

and

$$J_k(1) = \max_{(x_0,\dots,x_{k-1})} \ell_k(x_0,\dots,x_{k-1},1) \tag{A.5}$$

We can represent the problem in the form of a graph as in Figure A.1, with N stages where the nodes at the k-th stage represent the state x_k (either "0" or "1"), and the branches represent the *gain* associated to the transition from one state to the next. In this way $J_k(0)$ and $J_k(1)$ represent the maximum of all possible gains up to $x_k = 0$ and $x_k = 1$ respectively. These quantities can be recursively updated as

$$J_{k+1}(0) = \max\{J_k(0) + \Delta\ell_{k+1}(0,0)\ ,\ J_k(1) + \Delta\ell_{k+1}(1,0)\} \tag{A.6}$$

and

$$J_{k+1}(1) = \max\{J_k(0) + \Delta\ell_{k+1}(0,1)\ ,\ J_k(1) + \Delta\ell_{k+1}(1,1)\} \tag{A.7}$$

with $\Delta\ell_k$ as in Equation A.3. Also we can keep track of the branches which yield the maximum likelihood by defining $\text{Pointer}_k(0)$, and $\text{Pointer}_k(1)$ at each stage k. The sequence $\hat{x}$ maximizing $\ell_N(x_0,\dots,x_N)$ is therefore obtained by backtracking as in the following:

Let $\hat{x}_N$ be such that $J_N(\hat{x}_N) = \max\{J_N(0)\ ,\ J_N(1)\}$
for k = N - 1 to 0
$\hat{x}_k = \text{Pointer}_{k+1}(\hat{x}_{k+1})$
end for
end

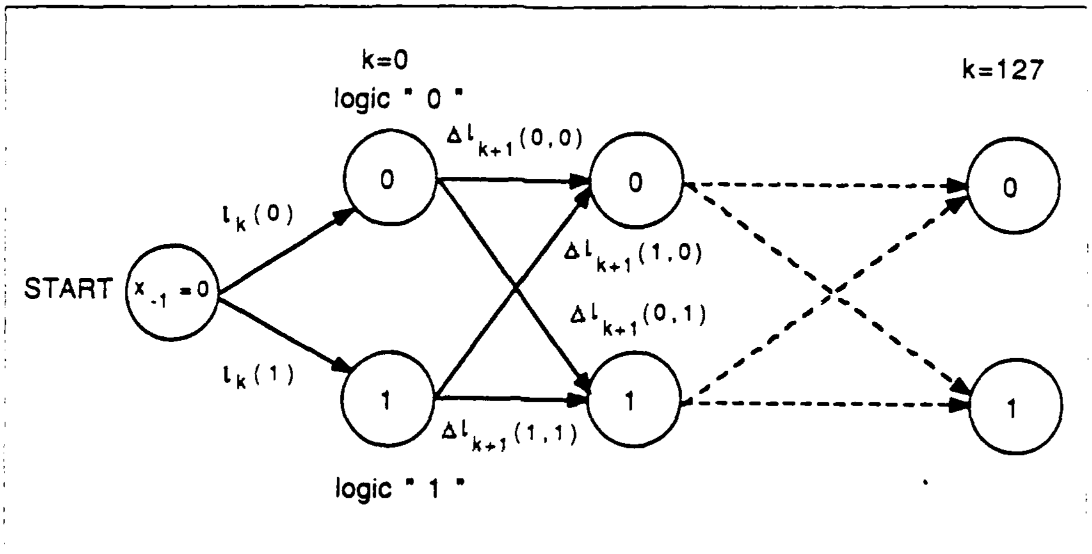


Figure A.1 Choice of Best Sequence of Decisions.

APPENDIX B
COMPUTER PROGRAMS

```
C       ************************************************************
C       *                                                          *
C       *      PROGRAM BINARY.DAT                                  *
C       *                                                          *
C       *      PURPOSE To generate 128 point binary sequence.      *
C       *                                                          *
C       *      REQUIRED IMSL ROUTINES None.                        *
C       *                                                          *
C       *      IMPLEMENTED BY Lt. Kani HACIPASAOGLU  Jan. 1987     *
C       *                                                          *
C       ************************************************************
C
        INTEGER INITT,FINITT,F(128),TERM,SETBUF,K
        REAL DWINDO,MOVEA,DRAWA
        OPEN(UNIT=1,NAME='BINARY.DAT',ACCESS='DIRECT',STATUS='NEW',
     *  RECL=32,MAXREC=128)
C
        DO 10 K=1,128
          IF(K .GE. 0 .AND. K .LT. 30) THEN
            F(K)=100
          ELSE IF(K .GT. 70 .AND. K .LT. 90) THEN
            F(K)=100
          ELSE IF(K .GT. 120 .AND. K .LE. 128) THEN
            F(K)=100
          ELSE
            F(K)=200
          END IF
       WRITE(1'K) (F(K))
 10    CONTINUE
        CLOSE(UNIT=1)
C
        CALL INITT(480)
        CALL TERM(2,1)
        CALL SETBUF(2)
        CALL DWINDO(-50.0,300.0,-50.0,400.0)
        CALL MOVEA(-50.0,-50.0)
        CALL DRAWA(300.0,-50.0)
        CALL DRAWA(300.0,400.0)
        CALL DRAWA(-50.0,400.0)
        CALL DRAWA(-50.0,-50.0)
        CALL MOVEA(0.0,1000.0)
        CALL DRAWA(0.0,0.0)
        CALL DRAWA(128.0,0.0)
        CALL MOVEA(0.0,0.0)
        DO 35 K=1,128
           X=K
           Y=F(K)
        CALL DRAWA(X,Y)
 35    CONTINUE
        CALL FINITT(0,0)
        STOP
        END
```

```
C       ****************************************************************
C       *                                                              *
C       *     PROGRAM BINARY2.DAT                                      *
C       *                                                              *
C       *     PURPOSE To generate one dimensional 128 point binary     *
C       *             sequence with zero mean Gaussian noise used      *
C       *             by Dynamic Programming algorithm to segment      *
C       *             a noisy signal.                                  *
C       *                                                              *
C       *     REQUIRED IMSL ROUTINES GGNML                             *
C       *                                                              *
C       *     IMPLEMENTED BY Lt. Kani HACIPASACGLU Feb. 1987           *
C       *                                                              *
C       ****************************************************************
C
        INTEGER INIIT,FINITT,F(128),TERM,SETBUF,I,NR,L
        REAL DWINDO,MOVEA,DRAWA,GGNML,R,S(128)
        DOUBLE PRECISION DSEED
        DSEED=65471
        OPEN(UNIT=2,NAME='BINARY2.DAT',ACCESS='DIRECT',STATUS='NEW',
     *  RECL=32,MAXREC=128)
C
        DO 10 I=1,128
          IF(I .GE. 0 .AND. I .LT. 30) THEN
            F(I)=100
          ELSE IF(I .GT. 70 .AND. I .LT. 90) THEN
            F(I)=100
          ELSE IF(I .GT. 120 .AND. I .LE.128) THEN
            F(I)=100
          ELSE
            F(I)=200
          END IF
        NR=1
        CALL GGNML(DSEED,NR,R)
        SIGMA=75.0
        R=SIGMA*R
C   R:NOISE FUNCTION WHICH WAS GENERATED
        S(I)=F(I)+R
        WRITE(2'I) (S(I))
 10     CONTINUE
        CLOSE(UNIT=2)
C
        CALL INITT(480)
        CALL TERM(2,1)
        CALL SETBUF(2)
        CALL DWINDO(-50.0,300.0,-50.0,400.0)
        CALL MOVEA(-50.0,-50.0)
        CALL DRAWA(300.0,-50.0)
        CALL DRAWA(300.0,400.0)
        CALL DRAWA(-50.0,400.0)
        CALL DRAWA(-50.0,-50.0)
        CALL MOVEA(0.0,1000.0)
        CALL DRAWA(0.0,0.0)
        CALL DRAWA(128.0,0.0)
        CALL MOVEA(0.0,0.0)
        DO 35 I=1,128
           X=I
           Y=S(I)
        CALL DRAWA(X,Y)
 35     CONTINUE
        CALL FINITT(0,0)
        STOP
        END
```

```
C       ***********************************************************************
C       *                                                                     *
C       *     PROGRAM ONUR2                                                   *
C       *                                                                     *
C       *     PURPOSE To segment a noisy binary sequence using                *
C       *             Dynamic Programming algorithm.                          *
C       *                                                                     *
C       *     REQUIRED IMSL ROUTINES None.                                    *
C       *                                                                     *
C       *     IMPLEMENTED BY Lt. Kani HACIPASAOGLU  March 1987                *
C       *                                                                     *
C       ***********************************************************************
C
        INTEGER N,K,F0,F1,OUT,PTR(128,0:1),F(128)
        INTEGER INITT,FINITT,TERM,SETBUF,J,M
        REAL BETA,SIGMA,LNEW0,LNEW1,MOVEA,L0,L1,OUTS(1:128),SUM.TEE,TOT
        REAL DELV(1:128,0:1,0:1),DWINDO,DRAWA,S(128),X,Y,SSE,SEE,TEE
         SIGMA=150.0
         READ(*,1) BETA
 1      FORMAT(F4.1)
         F0=100
         F1=200
        OPEN(UNIT=2,NAME='BINARY2.DAT',ACCESS='DIRECT',STATUS='OLD',
     *  RECL=32,MAXREC=128)
        DO 33 N=1,128
        READ (2'N) (S(N))
        WRITE(6,*) S(N),N
 33    CONTINUE
        CLOSE(UNIT=2)
        K=1
        L0=BETA-(((S(K)-100)**2)/(2*(SIGMA**2)))
        L1=-BETA-(((S(K)-200)**2)/(2*(SIGMA**2)))
        DO 15 M=1,128
         K=M-1
         DELV(K+1,0,0)=BETA-(((S(K+1)-F0)**2)/(2*(SIGMA**2)))
         DELV(K+1,1,0)=-BETA-(((S(K+1)-F0)**2)/(2*(SIGMA**2)))
         DELV(K+1,0,1)=-BETA-(((S(K+1)-F1)**2)/(2*(SIGMA**2)))
         DELV(K+1,1,1)=BETA-(((S(K+1)-F1)**2)/(2*(SIGMA**2)))
          IF((L0+DELV(K+1,0,0)) .LT. (L1+DELV(K+1,1,0))) THEN
            LNEW0=L1+DELV(K+1,1,0)
            PTR(M,0)=1
          ELSE
            LNEW0=L0+DELV(K+1,0,0)
            PTR(M,0)=0
         END IF
          IF((L0+DELV(K+1,0,1)) .LT. (L1+DELV(K+1,1,1))) THEN
            LNEW1=L1+DELV(K+1,1,1)
            PTR(M,1)=1
          ELSE
            LNEW1=L0+DELV(K+1,0,1)
            PTR(M,1)=0
          END IF
            L0=LNEW0
            L1=LNEW1
 15     CONTINUE
C   TRACE    BACK PROCEDURE
        OPEN(UNIT=3,NAME='OUTS.DAT',ACCESS='DIRECT',STATUS='NEW',
     *  RECL=32,MAXREC=128)
          IF(L0 .LT. L1) THEN
            J=1
          ELSE
            J=0
          END IF
          OUT=J
        DO 44 K=127,0,-1
         IF(OUT .EQ. 0) THEN
           OUTS(K+1)=100
         ELSE
           OUTS(K+1)=200
         END IF
```

```
       OUT=PTR(K+1,OUT)
       WRITE(6,*) OUTS(K+1),K+1
 44   CONTINUE
       DO 55 K=1,128
       WRITE(3'K)(OUTS(K))
 55   CONTINUE
      CLOSE(UNIT=3)
C
      CALL INITT(480)
      CALL TERM(2,1)
      CALL SETBUF(2)
      CALL DWINDO(-50.0,300.0,-50.0,400.0)
      CALL MOVEA(-50.0,-50.0)
      CALL DRAWA(300.0,-50.0)
      CALL DRAWA(300.0,400.0)
      CALL DRAWA(-50.0,400.0)
      CALL DRAWA(-50.0,-50.0)
      CALL MOVEA(0.0,1000.0)
      CALL DRAWA(0.0,0.0)
      CALL DRAWA(150.0,0.0)
      CALL MOVEA(0.0,0.0)
      DO 3 K=1,128
        X=K
        Y=OUTS(K)
     CALL DRAWA(X,Y)
 3   CONTINUE
     CALL FINITT(0,0)
     STOP
     END
```

```
C      ************************************************************
C      *                                                          *
C      *     PROGRAM  TEST.DAT                                    *
C      *                                                          *
C      *     PURPOSE  To generate a test image which has          *
C      *              three different intensity levels.           *
C      *                                                          *
C      *     REQUIRED IMSL ROUTINES None.                         *
C      *                                                          *
C      *     IMPLEMENTED BY Lt. Kani HACIPASAOGLU  May 1987       *
C      *                                                          *
C      ************************************************************
C
        BYTE A(128,128)
        INTEGER R,XC1,YC1,XC2,YC2,F1(1:128,1:128),F2(1:128,1:128)
C
        R=20
        XC1=60
        YC1=60
        YC2=60
        XC2=90
C
        OPEN(UNIT=5,NAME='TEST.DAT',ACCESS='DIRECT',STATUS='NEW',
     *  RECL=32,MAXREC=128)
C
        DO 10 I=1,128
         DO 20 J=1,128
          F1(I,J)=(((I-XC1)**2+(J-YC1)**2)-(R**2))
          F2(I,J)=(((I-XC2)**2+(J-YC2)**2)-(R**2))
        IF((F1(I,J) .LE. 0) .AND. (F2(I,J) .GT. 0)) THEN
          A(I,J)=50
        ELSE IF((F1(I,J) .GT. 0) .AND. (F2(I,J) .LE. 0)) THEN
          A(I,J)=150
        ELSE IF((F1(I,J) .LE. 0) .AND. (F2(I,J) .LE. 0)) THEN
          A(I,J)=100
        ELSE
          A(I,J)=200
        END IF
  20   CONTINUE
        WRITE(5'I) (A(I,J),J=1,128)
  10   CONTINUE
        CLOSE(UNIT=5)
        STOP
        END
```

```
C      ******************************************************
C      *                                                    *
C      *     PROGRAM  TESTIM.DAT                            *
C      *                                                    *
C      *     PURPOSE  To generate a test image with zero mean *
C      *              Gaussian noise to be segmented by      *
C      *              Kalman Filter.                         *
C      *                                                    *
C      *     REQUIRED IMSL ROUTINES GGNML                   *
C      *                                                    *
C      *     IMPLEMENTED BY Lt. Kani HACIPASAOGLU  May 1987  *
C      *                                                    *
C      ******************************************************
C
        BYTE A(128,128)
        CHARACTER*50 TESTIM
        INTEGER XC1,YC1,XC2,YC2,F1(128,128)
        INTEGER NR,Rr,SIGMA,CR,F2(128,128),AA(128,128)
        REAL GGNML,R
        DOUBLE PRECISION DSEED
        DSEED=65471
        WRITE(*,100)
  100  FORMAT('ENTER IMAGE DATA NAME')
        READ(*,101)
  101  FORMAT(A50)
C
        CR=20
        XC1=60
        YC1=60
        YC2=60
        XC2=90
C
        OPEN(UNIT=1,NAME='TESTIM.DAT',ACCESS='DIRECT',STATUS='NEW',
      *  RECL=32,MAXREC=128)
C
        WRITE(*,222)
 222   FORMAT('ENTER SIGMA')
        READ(*,333) SIGMA
 333   FORMAT(I2)
        DO 10 I=1,128
         DO 20 J=1,128
          F1(I,J)=(((I-XC1)**2+(J-YC1)**2)-(CR**2))
          F2(I,J)=(((I-XC2)**2+(J-YC2)**2)-(CR**2))
        IF((F1(I,J) .LE. 0) .AND. (F2(I,J) .GT. 0)) THEN
          A(I,J)=50
        ELSE IF((F1(I,J) .GT. 0) .AND. (F2(I,J) .LE. 0)) THEN
          A(I,J)=150
        ELSE IF((F1(I,J) .LE. 0) .AND. (F2(I,J) .LE. 0)) THEN
          A(I,J)=100
        ELSE
          A(I,J)=200
        END IF
        NR=1
        CALL GGNML(DSEED,NR,R)
        Rr=SIGMA*jnint(R)
        AA(I,J)=A(I,J)+Rr
        IF(AA(I,J) .LT. -128) THEN
          AA(I,J)=AA(I,J)+256
        ELSE IF(AA(I,J) .GT. 127) THEN
          AA(I,J)=AA(I,J)-256
        ELSE
          AA(I,J)=AA(I,J)
        END IF
        A(I,J)=AA(I,J)
 20    CONTINUE
        WRITE(1'I) (A(I,J),J=1,128)
 10    CONTINUE
        CLOSE(UNIT=1)
        STOP
        END
```

```
C       ***************************************************************
C       *                                                             *
C       *      PROGRAM   FINAL.DAT                                    *
C       *                                                             *
C       *      PURPOSE   To generate a test image which has 16        *
C       *                different regions.                           *
C       *                                                             *
C       *      REQUIRED IMSL ROUTINES None.                           *
C       *                                                             *
C       *      IMPLEMENTED BY Lt. Kani HACIPASAOGLU  June 1987        *
C       *                                                             *
C       ***************************************************************
C
        BYTE A(128,128)
C
        OPEN(UNIT=1,NAME='FINAL.DAT',ACCESS='DIRECT',STATUS='NEW',
     *  RECL=32,MAXREC=128)
        DO 10 K=1,32
          DO 20 L=1,32
              A(K,L)=100
              A(K,L+32)=50
              A(K,L+64)=180
              A(K,L+96)=70
C
              A(K+32,L)=200
              A(K+32,L+32)=120
              A(K+32,L+64)=60
              A(K+32,L+96)=140
C
              A(K+64,L)=75
              A(K+64,L+32)=175
              A(K+64,L+64)=90
              A(K+64,L+96)=65
C
              A(K+96,L)=130
              A(K+96,L+32)=55
              A(K+96,L+64)=190
              A(K+96,L+96)=110
 20        CONTINUE
 10     CONTINUE
        DO 30 I=1,128
          WRITE(1'I) (A(I,J),J=1,128)
 30     CONTINUE
        CLOSE(UNIT=1)
        STOP
        END
```

```
C       *****************************************************************
C       *                                                               *
C       *     PROGRAM  NFINAL.DAT                                       *
C       *                                                               *
C       *     PURPOSE  To generate a test image which has 16            *
C       *              different regions with zero mean Gaussian        *
C       *              noise to be segmented by Kalman Filter.          *
C       *                                                               *
C       *     REQUIRED IMSL ROUTINES GGNML                              *
C       *                                                               *
C       *     IMPLEMENTED BY Lt. Kani HACIFASAOGLU  June 1987           *
C       *                                                               *
C       *****************************************************************
C
        BYTE A(128,128)
        CHARACTER*50 NFINAL
        INTEGER NR,Rr,SIGMA,AA(128,128)
        REAL    GGNML,R
        DOUBLE PRECISION DSEED
        DSEED=65471
        WRITE(*,100)
  100  FORMAT( ' ENTER IMAGE DATA NAME')
        READ(*,101)
  101  FORMAT(A50)
C
        OPEN(UNIT=1,NAME='NFINAL3.DAT',ACCESS='DIRECT',STATUS='NEW',
     *  RECL=32,MAXREC=128)
        WRITE(*,222)
  222  FORMAT( ' ENTER SIGMA')
        READ(*,333) SIGMA
  333  FORMAT(I2)
        DO 10 K=1,32
          DO 20 L=1,32
              A(K,L)=100
              A(K,L+32)=50
              A(K,L+64)=180
              A(K,L+96)=70
C
              A(K+32,L)=200
              A(K+32,L+32)=120
              A(K+32,L+64)=60
              A(K+32,L+96)=140
C
              A(K+64,L)=75
              A(K+64,L+32)=175
              A(K+64,L+64)=90
              A(K+64,L+96)=65
C
              A(K+96,L)=130
              A(K+96,L+32)=55
              A(K+96,L+64)=190
              A(K+96,L+96)=110
 20      CONTINUE
 10     CONTINUE
        NR=1
        DO 80 I=1,128
           DO 90 J=1,128
              CALL GGNML(DSEED,NR,R)
              Rr=SIGMA*jnint(R)
              AA(I,J)=A(I,J)+Rr
              IF(AA(I,J) .LT. -128) THEN
                AA(I,J)=AA(I,J)+256
              ELSE IF(AA(I,J) .GT. 127) THEN
                AA(I,J)=AA(I,J)-256
              ELSE
                AA(I,J)=AA(I,J)
              END IF
              A(I,J)=AA(I,J)
  90       CONTINUE
           WRITE(1'I) (A(I,J),J=1,128)
```

```
80     CONTINUE
       CLOSE(UNIT=1)
       STOP
       END
```

```
c       ***********************************************************
c       *                                                         *
c       *     PROGRAM  CRISTI2D                                   *
c       *                                                         *
c       *     PURPOSE  To segment a two dimensional noisy image   *
c       *              using Markov Random Field, Dynamic         *
c       *              Programming and Kalman Filtering techniques. *
c       *                                                         *
c       *     REQUIRED IMSL ROUTINES None.                        *
c       *                                                         *
c       *     IMPLEMENTED BY Lt. Kani HACIPASAOGLU  Aug. 1987     *
c       *                                                         *
c       ***********************************************************
c
          common ny, y(512,512),tau(512),xtop(512),iedge,ymax
          integer start, dir, cycle
c
          character*20 image, imout
          byte s(512,512), v(512,512)
          write(*, 777)
777     format ('   Enter Input Image File Name:')
          read (*, 778) image
          write(*, 782)
782     format('   Enter Output Image File Name:')
          read(*, 778) imout
778     format(a20)
          open (2, name=image, access='direct', status='old')
c       get data from file
          write (*, 779)
779     format('  Enter Image Size')
          read(*, 780) npoints
780     format(i3)
          write(*, 555)
          read (*, 666) sigma, beta
          write (*, 790)
790     format('  see edges? (yes=1)')
          read (*, 791) iedge
791     format(i1)
          write(*, 810)
          write(*, 792)
792     format('   Enter scale factor (1.0=no scale)')
810     format('  output yout=yth+scale*(y-yth)')
          read(*, 793) scale
793     format(f10.2)
          write(*,811)
811     format(' 1: yth=(yav+ymax)/2; 2:yth=0.9ymax; 3: enter yth;  ??')
          read(*, 791) iyth
          if (iyth.eq.3) then
          write(*, 794)
794     format('  Enter yth:')
          read (*, 793) yth
          endif
c
          do 5 k=1, npoints
             read(2'k) (S(k,j), j=1, npoints)
                do 6 j=1, npoints
                     temp=s(k,j)
                     if (temp.lt.0.0) then
                     temp=temp+256
                     endif
                     y(k,j)=temp
6           continue
5       continue
          close (unit=2)
c
c
555     format('   ENTER: sigma, beta')
666     format(2f10.4)
c
c       *** main loop ***
```

```
        write(*, 667)
667     format('  Start Computing ...')
        ymax=0.0
        yav=0.0
        ncount=0
        nx=npoints
        ny=npoints
        gam=1.0/(2.0*sigma**2)
c
c       *** initialize
c
        do 30 j=1,ny
        tau(j)=0.0
        xtop(j)=0.0
30      continue
c
        nx=nx-1
c       *** main loop  ***
        do 40 i=1,nx
        start=1
        dir=+1
        cycle=1
        call line(i, start, dir, cycle, gam, beta, rowav)
c
        start=ny
        dir=-1
        cycle=2
        call line(i,start, dir, cycle, gam, beta,rowav)
40      continue
c
c       *** compute average of output image
        yav=(ncount*yav+rowav)/(ncount+1)
        ncount=ncount+1
c
        write(*, 668)
668     format('  ... End Computing. Now transfering data to disk ...')
c
c       *** output data ***
c
           if (iyth.eq.2) then
             yth=ymax*0.9
                endif
           if (iyth.eq.1) then
             yth=(yav+ymax)/2.0
                endif
c
        write(*, 800)
800     format('  yave, ymax, yth =')
        write(*, 801) yav, ymax, yth
801     format(3(x, f10.4))

        do 60 k=1, npoints
           do 61 j=1, npoints
              ys=yth+scale*(y(k,j)-yth)
c
                   if (ys.gt.255.0) then
                     ys=255.0
                   endif
                   if (ys.le.0.0) then
                     ys=0.0
                   endif
c
                 itemp=jnint(ys)
                 if (itemp.gt.127) then
                 itemp=itemp-256
                 endif
             v(k,j)=itemp
61      continue
60      continue
```

```
c
          open (unit=3, name=imout, access='direct',
     *     status='new', recl=128, maxrec=512)
c
          do 50 j=1, npoints
          write(3'j) (v(j,k), k=1,npoints)
50      continue
          close (unit=3)
c
          stop
          end
c
          subroutine line(i, start, dir, cycle, gam, beta,rowav)
          integer start, dir, cycle
          common ny, y(512,512), tau(512), xtop(512),iedge,ymax
          real xhat(4,512), lambda(4,512)
          real ttau(4), txtop(4), tlambda(4), txhat(4), te(4)
          real newtau(4,512), newxtop(4,512)
          real e(4,512)
          integer pointer(4,512)
c
c       *** scan from START by DIR (=+1, -1)
          j=start
c       *** initialize first column
          do 90 k=1,4
          xhat(k,start)=y(i,start)
          lambda(k,start)=0.0
          e(k,start)=0.0
90      continue
c
c
          rowav=0.0
          ncount=0
999     j=j+dir
c       *** state=(left edge, upper edge)
c       *** 0=no edge, 1=edge
c
c       *** state (0,0)
          do 110 k=1,4
          ttau(k)=tau(j)+1
          txtop(k)=xtop(j)+(1.0/ttau(k))*(y(i,j)-xtop(j))
          tlambda(k)=lambda(k,j-dir)+ttau(k)
              z=txtop(k)-xhat(k,j-dir)
          txhat(k)=xhat(k,j-dir)+(ttau(k)/tlambda(k))*z
c
          e1=(y(i,j+dir)-txhat(k))**2
          e2=(y(i,j)-txhat(k))**2
          e3=(y(i+1,j)-txhat(k))**2
          te(k)=e(k,j-dir)+gam*(e1+e2+e3)/3 - 2*beta
c
          if (k.eq.1) then
          emin=te(k)
          endif
          if (te(k).le.emin) then
          im=k
          emin=te(k)
          endif
110     continue
c
          newtau(4,j)=txtop(im)
          lambda(4,j)=tlambda(im)
          pointer(4,j)=im
          newxtop(4,j)=txtop(im)
          xhat(4,j)=txhat(im)
          e(4,j)=te(im)
c
c       *** state (0,1) ***
          do 115 k=1,4
          tlambda(k)=lambda(k, j-dir)+1
              z=y(i,j)-xhat(k,j-dir)
```

```
          txhat(k)=xhat(k,j-dir)+(1.0/tlambda(k))*z
c
          e1=(y(i,j+dir)-txhat(k))**2
          e2=(y(i,j)-txhat(k))**2
          e3=(y(i+1,j)-txhat(k))**2
          te(k)=e(k,j-dir)+gam*(e1+e2+e3)/3
c
          if (k.eq.1) then
                    emin=te(k)
                    endif
          if (te(k).le.emin) then
                    im=k
                    emin=te(k)
                    endif
115     continue
c
          newtau(1,j)=1
          newxtop(1,j)=y(i,j)
          lambda(1,j)=tlambda(im)
          pointer(1,j)=im
          xhat(1,j)=txhat(im)
          e(1,j)=te(im)
c
c
c       *** state (1,0)
          do 120 k=1,4
          ttau(k)=tau(j)+1
          txtop(k)=xtop(j) +  (1.0/ttau(k))*(y(i,j)-xtop(j))
c
          e1=(y(i,j+dir)-txtop(k))**2
          e2=(y(i,j)-txtop(k))**2
          e3=(y(i+1,j)-txtop(k))**2
          te(k)=e(k,j-dir) + gam*(e1+e2+e3)/3
c
          if (k.eq.1) then
                    emin=te(k)
                    endif
          if (te(k).le.emin) then
                    im=k
                    emin=te(k)
                    endif
120     continue
c
c
          newtau(2,j)=ttau(im)
          newxtop(2,j)=txtop(im)
          lambda(2,j)=tau(j)
          pointer(2,j)=im
          xhat(2,j)=txtop(im)
          e(2,j)=te(im)
c
c
c       *** state (1,1)
          newtau(3,j)=1
          newxtop(3,j)=y(i,j)
          lambda(3,j)=1
          xhat(3,j)=y(i,j)
          e1=(y(i,j+dir)-y(i,j))**2
          e3=(y(i+1,j)-y(i,j))**2
          z2=gam*(e1+e3)/3
          if (e(4, j-dir).lt.e(1,j-dir)) then
                    e(3,j)=z2+e(4,j-dir)+beta
                    pointer(3,j)=4
          else
                    e(3,j)=z2+e(1,j-dir)+beta
                    pointer(3,j)=1
          endif
c
c
          if (j.gt.(1-dir).and.j.lt.(ny-dir)) then
```

```
                    goto 999
                    endif
c
c       *** backtrack ***
c
          emin=e(1,j)
          do 150 k=1,4
          if (emin.le.e(k,j)) then
                    emin=e(k,j)
                    im=k
                    endif
150     continue
c
888     y(i,j)=xhat(im,j)
c
c       write(*, 111) im
111     format(i3)
c
          if (cycle.eq.2) then
                    xtop(j)=newxtop(im,j)
                    tau(j)=newtau(im,j)
c
c       *** compute max intensity
            if (y(i,j).gt.ymax) then
                ymax=y(i,j)
            endif
c
c       *** row average
            rowav=(ncount*rowav+y(i,j))/(ncount+1)
            ncount=ncount+1
c
c           mark the edge with a black entry
                        if (im.ne.4.and.iedge.eq.1) then
                        y(i,j)=0.0
                        endif
                    endif
c
          im=pointer(im,j)
          j=j-dir
          if (j.gt.(1-dir).and.j.lt.(ny-dir)) then
                    goto 888
          endif
c
          return
          end
```

LIST OF REFERENCES

1. Derin, H., Elliott, H., Cristi, R. and Geman, D., *Bayes Smoothing Algorithms for Segmentation of Binary Images Modeled by Markov Random Fields*, IEEE Transactions on Pattern Analysis and Machine Intelligence, v. PAMI-6, pp. 707-720, November 1984.

2. Derin, H. and Elliott, H., *Modeling and Segmentation of Noisy and Textured Images Using Gibbs Random Fields*, IEEE Transactions on Pattern Analysis and Machine Intelligence, v. PAMI-9, pp. 39-55, January 1987.

3. Geman, S. and Geman, D., *Stochastic Relaxation, Gibbs Distributions and the Bayesian Restoration of Images*, IEEE Transactions on PAMI, v. PAMI-6, pp. 721-741, November 1984.

4. Besag, J., *On the Statistical Analysis of Dirty Pictures*, J. Royal Statistical Soc. B, no 3, pp. 259-302, 1986.

5. Therrien, C. W., *An Estimation-Theoretic Approach to Terrain Image Segmentation*, Computer Vision, Graphics and Image Processing, v. 22, pp. 313-326, June 1983.

6. Besag, J., *Spatial Interaction and the Statistical Analysis of Lattice Systems*, J. Royal Stat. Soc. B, v. 36, pp. 192-236, 1974.

7. Elliot, H., Derin, H., Cristi, R. and Geman, D., *Application of the Gibbs Distribution to Image Segmentation*, Proc. 1984 IEEE ASSP, pp. 32.5.1-32.5.4, San Diego, CA, March 1984.

8. Kinderman, R. and Snell, J. L., *Markov Random Fields and Their Applications*, American Math. Soc., Providence RI, 1980.

9. Bellman, R. E. and Dreyfus, S. E., *Applied Dynamic Programming*, Princeton Univ. Press, 1962.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Department Chairman, Code 62 Naval Postgraduate School Monterey, CA 93943-5000	3
4.	Roberto Cristi, Code 62CX Naval Postgraduate School Monterey, CA 93943-5000	3
5.	Ralph Hippenstiel, Code 62HI Naval Postgraduate School Monterey, CA 93943-5000	1
6.	Kani Hacipasaoglu Bascavus Sokak Birlik Apt. No. :32 10 Seyranbaglari Ankara, TURKEY	2
7.	Kara Kuvvetleri Komutanligi Personel Egitim Daire Baskanligi Bakanliklar - Ankara, TURKEY	3
8.	Kara Harp Okulu Ogretim Kurulu Baskanligi Bakanliklar - Ankara, TURKEY	2
9.	O. D. T. U. Kutuphanesi Orta Dogu Teknik Universitesi Ankara, TURKEY	1

END
DATE
FILMED
8-88
DTIC